Universidad Popular del Cesar

Minería de datos y Patrones

Version 2024-1

Local Binary Patterns

[Capítulo 2]

Dr. José Ramón Iglesias
DSP-ASIC BUILDER GROUP
Director Semillero TRIAC
Ingenieria Electronica
Universidad Popular del Cesar

- Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.
- Due to its discriminative power and computational simplicity, LBP texture operator has become a popular approach in various applications

Concept

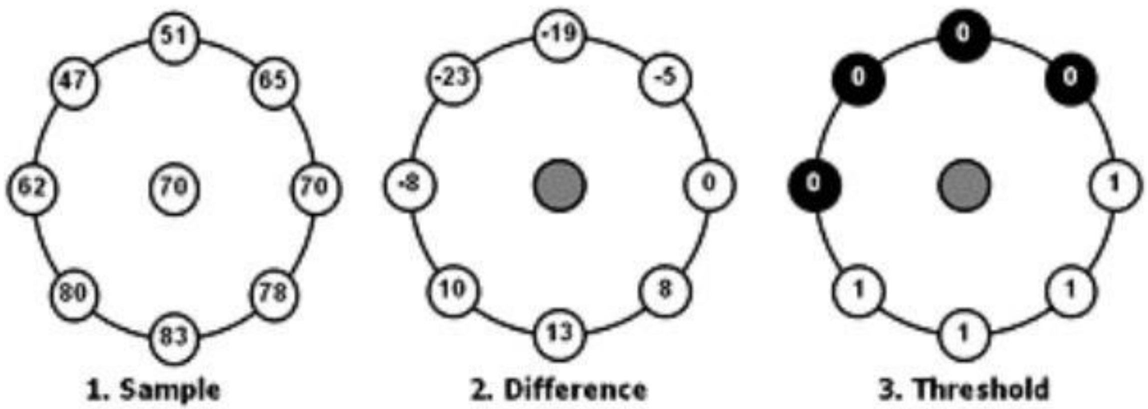
- Where the center pixel's value is greater than the neighbor, write "1". Otherwise, write "0". This gives an 8-digit binary number (which is usually converted to decimal for convenience).
- Compute the histogram, over the cell, of the frequency of each "number" occurring (i.e., each combination of which pixels are smaller and which are greater than the center).

Concept

- Illustration

The value of the LBP code of a pixel (x_c, y_c) is given by:

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p \qquad s(x) = \begin{cases} 1, if\ x \geq 0; \\ 0, otherwise. \end{cases}$$



1*1 + 1*2 + 1*4 + 1*8 + 0*16 + 0*32 + 0*64 + 0*128 = **15**

4. Multiply by powers of two and sum

Pixel Neighborhood-based Feature

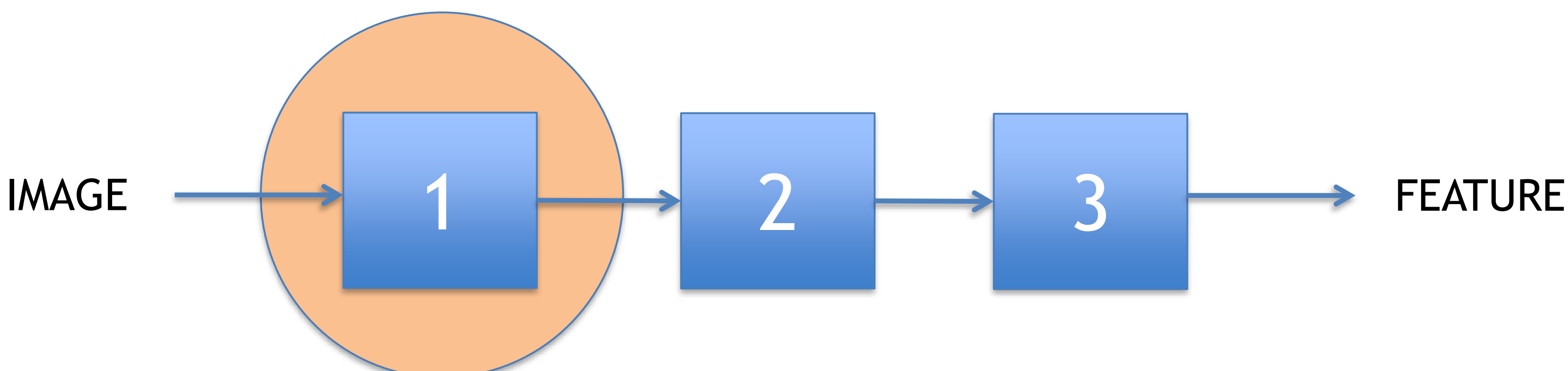
- The most important for texture analysis is to describe the spatial behavior of intensity values in any given neighborhood.
- Different methodologies have been proposed.
- Local binary pattern (LBP) is one of the most-widely used approach - mainly for face recognition.
- LBP is used for texture analysis too.

Local Binary Patterns

1. Coding
2. Mapping
3. Histogram

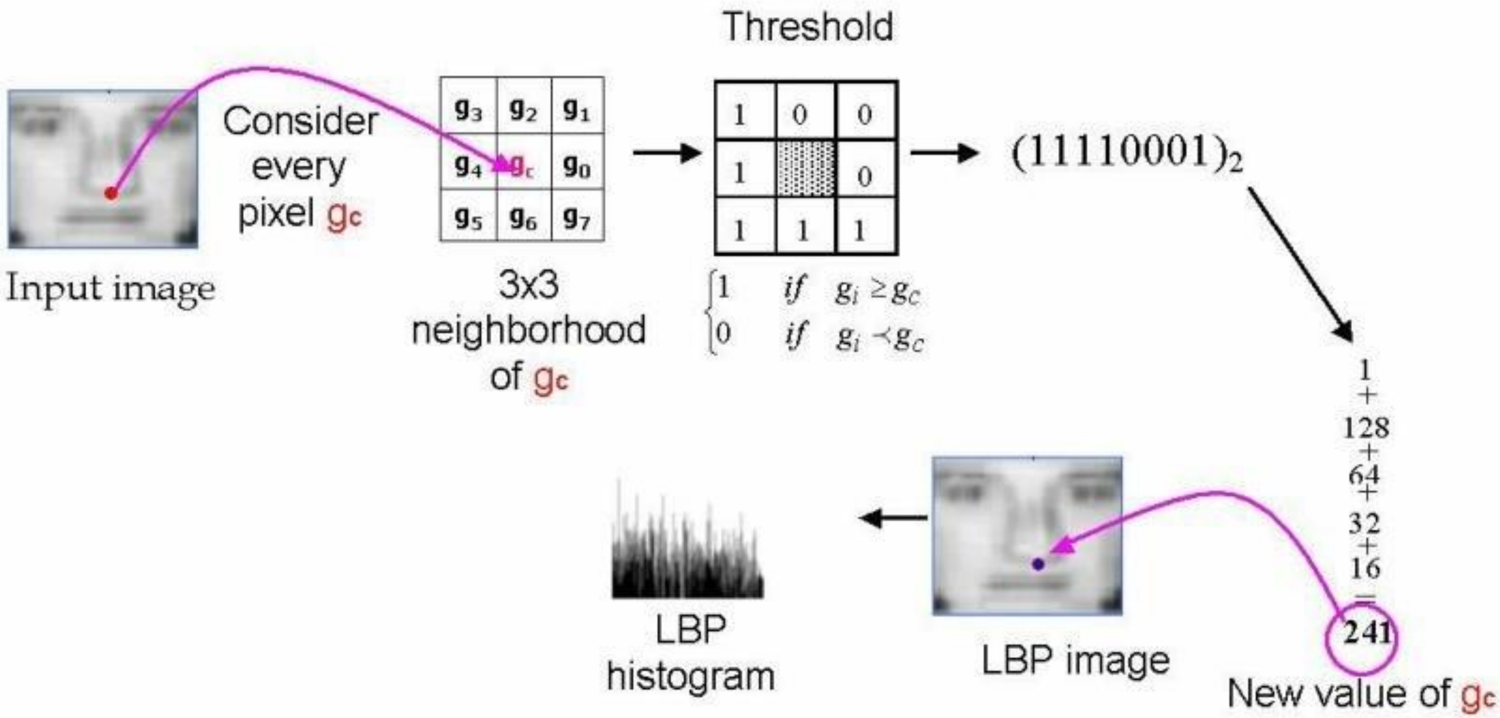
Local Binary Patterns

1. Coding
2. Mapping
3. Histogram



LBP: Example

- Local Binary Pattern (LBP) is a texture descriptor which codifies local primitives (such as curved edges, spots, flat areas, etc.) into a feature histogram.



4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9
9	6	4
9	6	2

<		

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9
9	6	4
9	6	2

0: <
1: ≥

0		

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9
9	6	4
9	6	2

0	≥	

0: <
1: ≥

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9
9	6	4
9	6	2

0	1	

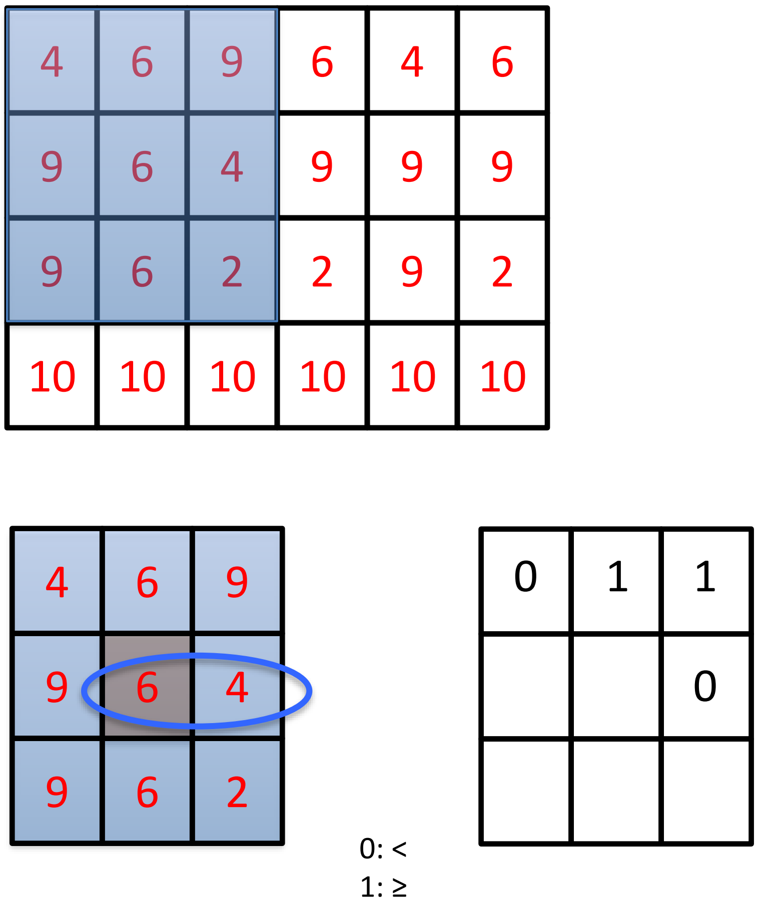
0: <
1: ≥

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9
9	6	4
9	6	2

0	1	1

0: <
1: ≥



4 6 9 6 4 6
9 6 4 9 9 9
9 6 2 2 9 2
10 10 10 10 10 10
4 6 9
9 6 4
9 6 2
0 1 1
0
0: <
1: ≥

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9
9	6	4
9	6	2

0	1	1
		0
		0

0: <
1: ≥

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9
9	6	4
9	6	2

0	1	1
		0
	1	0

0: <

1: ≥

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9
9	6	4
9	6	2

0	1	1
		0
1	1	0

0: <

1: ≥

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9
9	6	4
9	6	2

0	1	1
1		0
1	1	0

0: <

1: ≥

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9
9	6	4
9	6	2

0: <
1: ≥

0	1	1
1		0
1	1	0

x

1	2	4
128	+	8
64	32	16

= 2+4+32+64+128 = 230

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

4	6	9
9	6	4
9	6	2

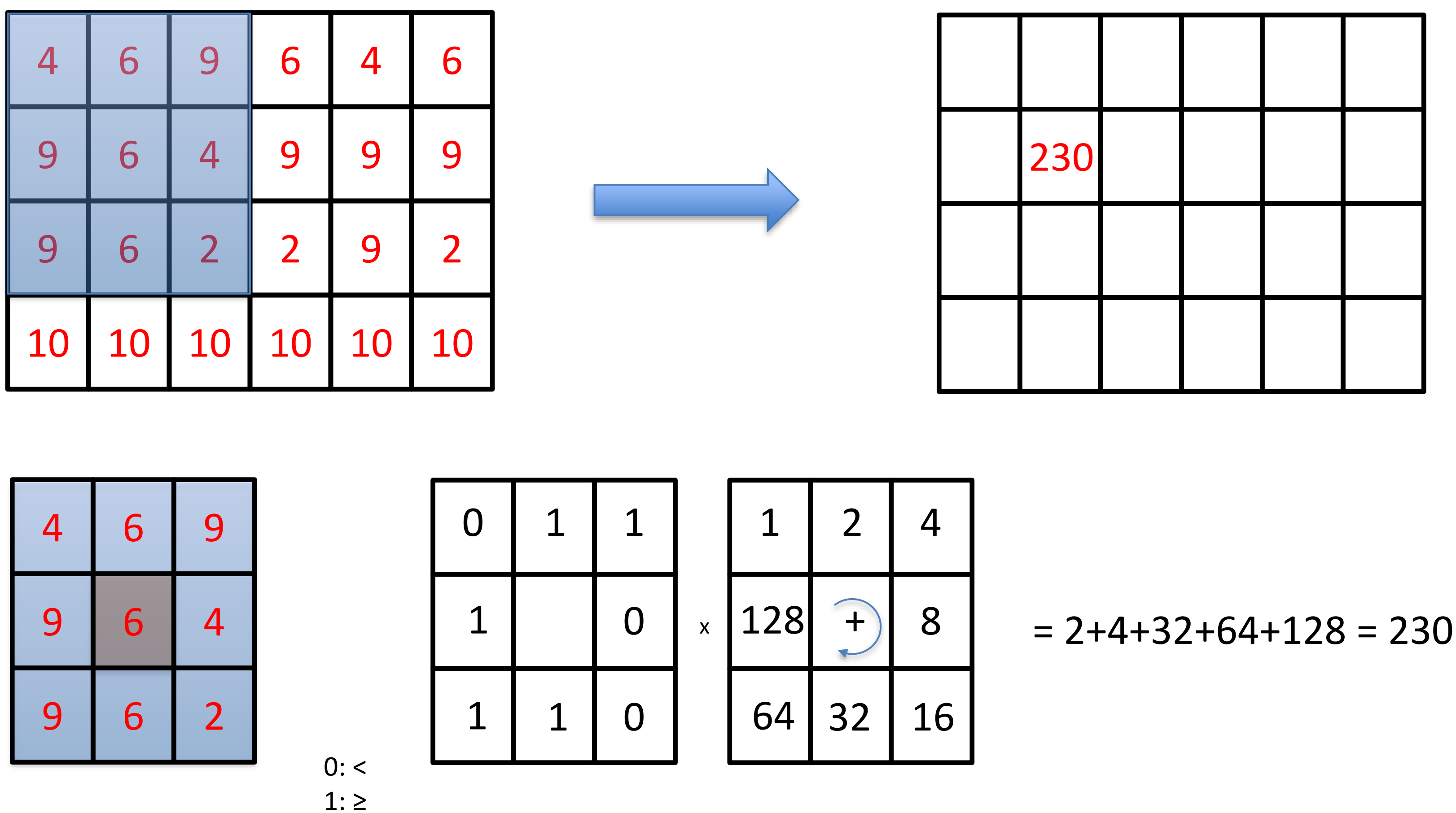
0: <
1: ≥

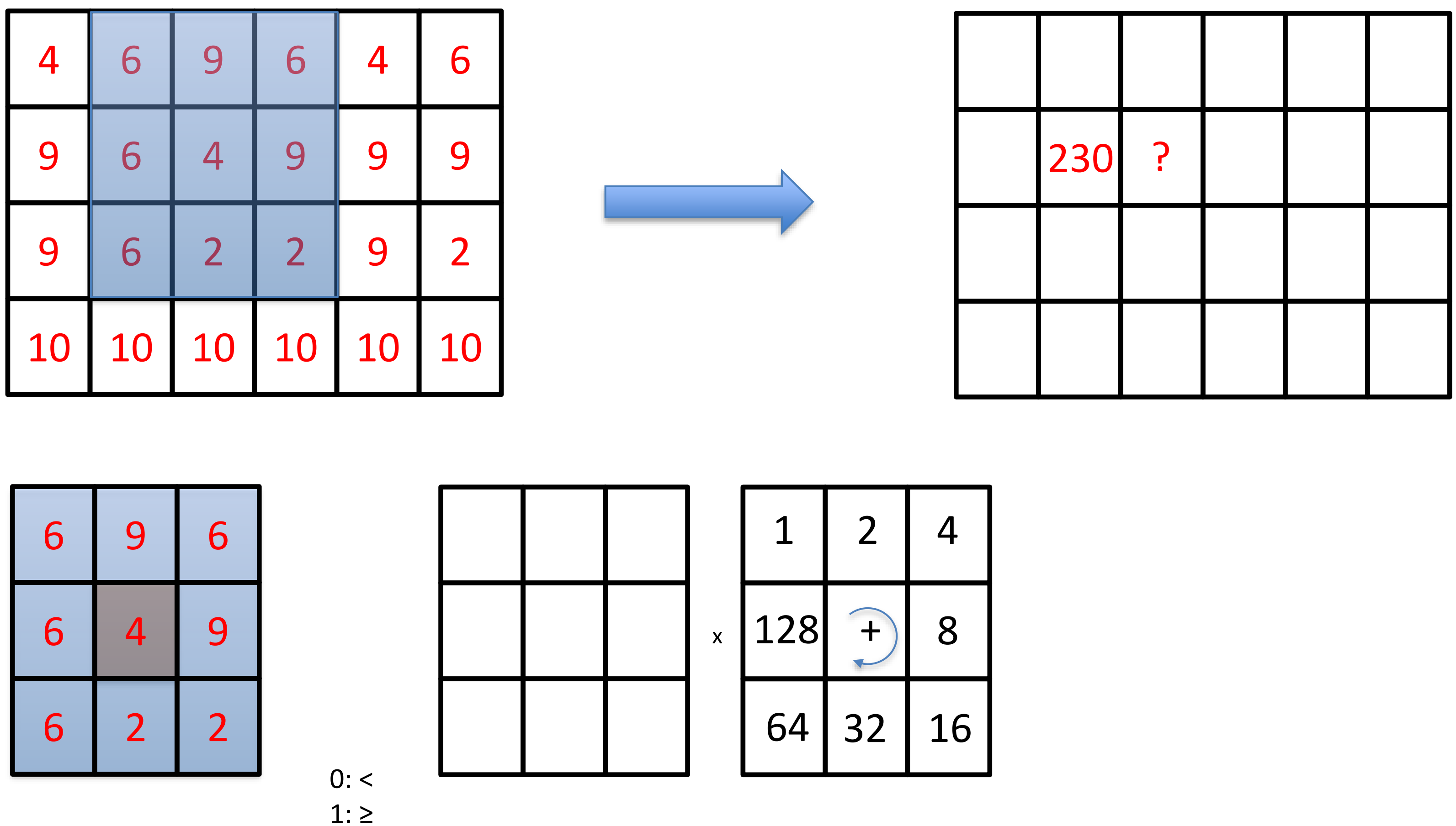
0	1	1
1		0
1	1	0

x

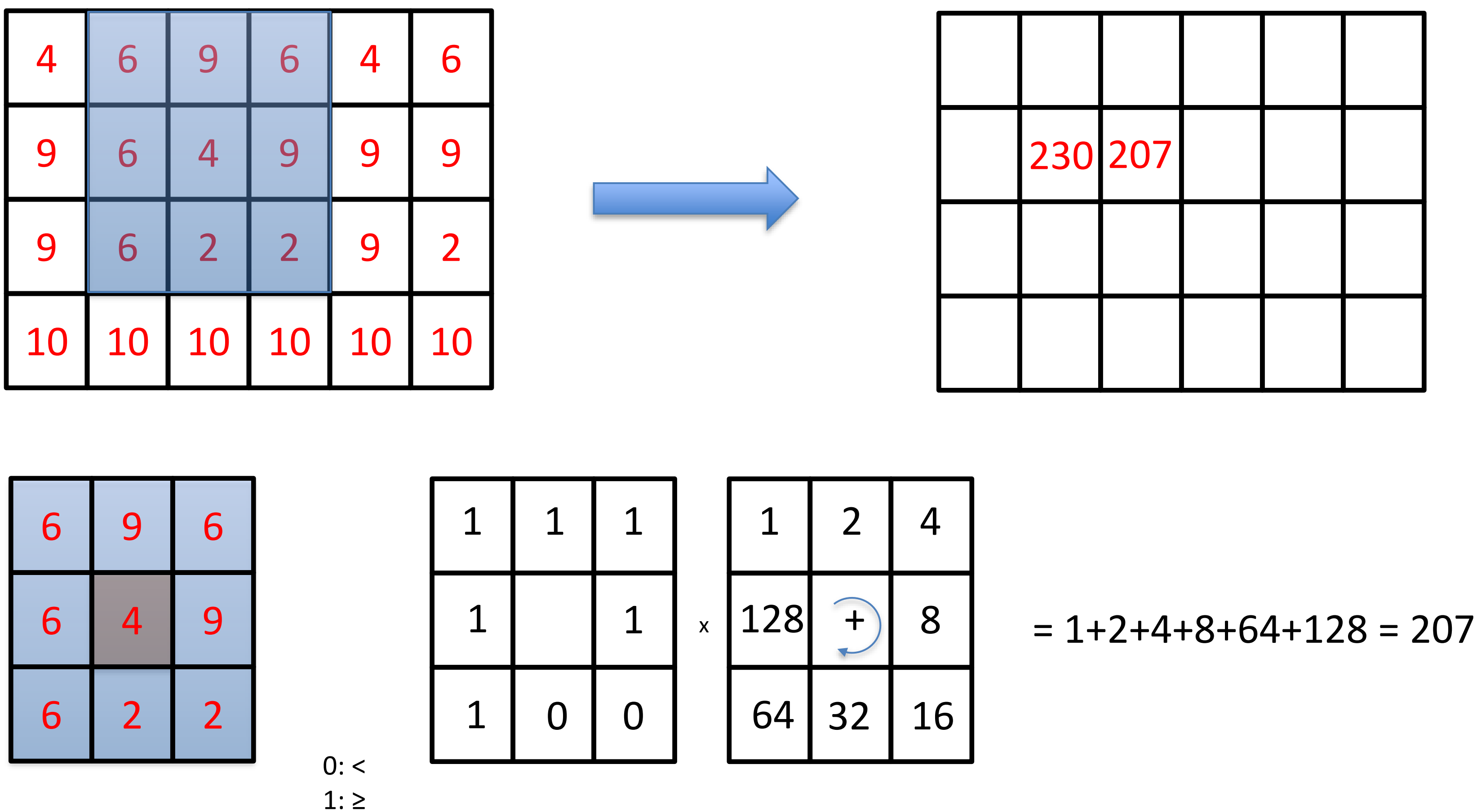
1	2	4
128	+	8
64	32	16

= 2+4+32+64+128 = 230


4 6 9 6 4 6
9 6 4 9 9 9
9 6 2 2 9 2
10 10 10 10 10 10
230
4 6 9
9 6 4
9 6 2
0: <
1: ≥
0 1 1
1 0
1 1 0
x
1 2 4
128 + 8
64 32 16
= 2+4+32+64+128 = 230



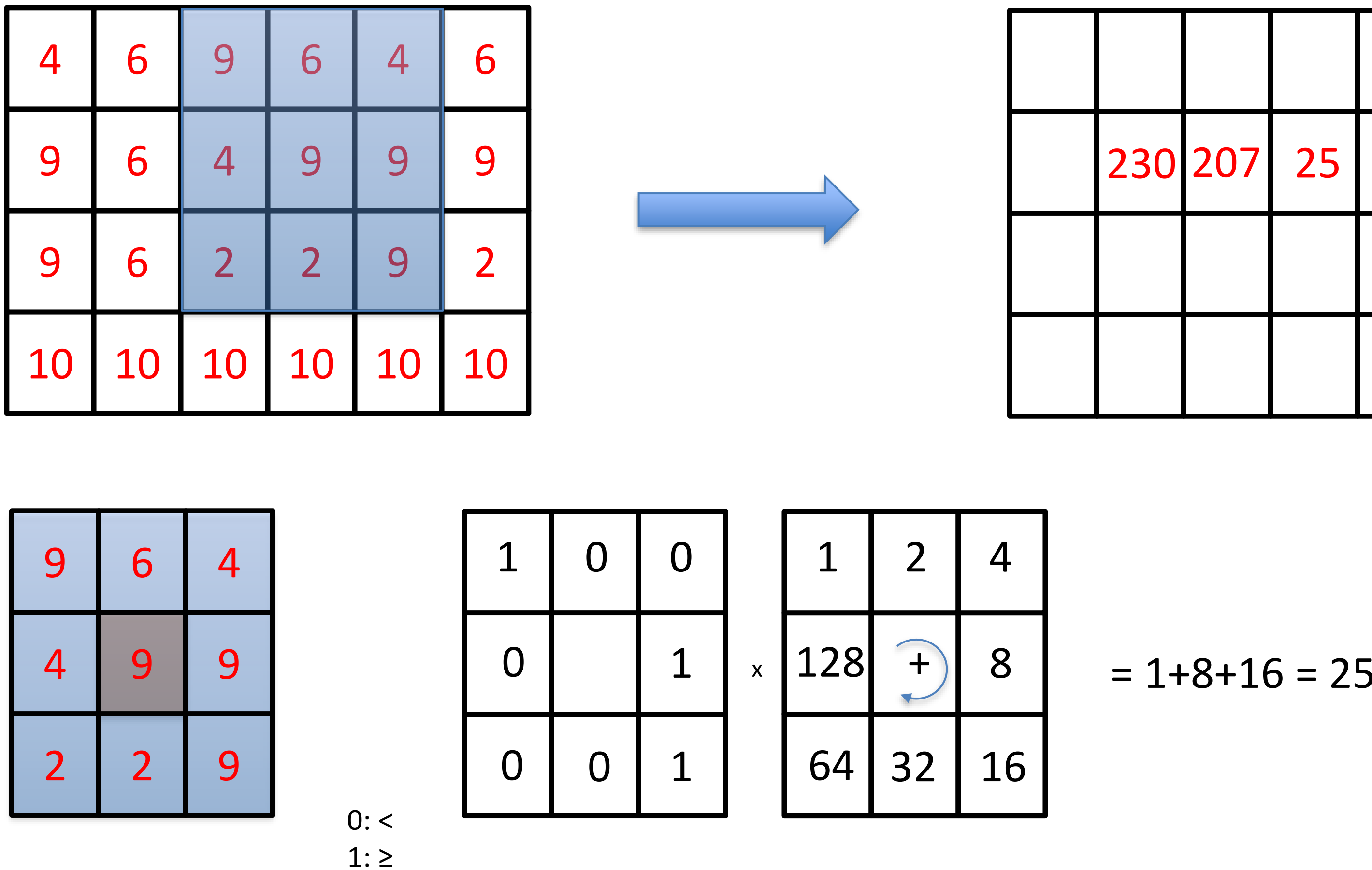
4 6 9 6 4 6
9 6 4 9 9 9
9 6 2 2 9 2
10 10 10 10 10 10
230 ?
6 9 6
6 4 9
6 2 2
0: <
1: ≥
x
1 2 4
128 + 8
64 32 16



4 6 9 6 4 6
9 6 4 9 9 9
9 6 2 2 9 2
10 10 10 10 10 10
230 207
6 9 6
6 4 9
6 2 2
0: <
1: ≥
1 1 1
1 1
1 0 0
x
1 2 4
128 + 8
64 32 16
= 1+2+4+8+64+128 = 207

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

	230	207	?		


4 6 9 6 4 6
9 6 4 9 9 9
9 6 2 2 9 2
10 10 10 10 10 10
230 207 25
9 6 4
4 9 9
2 2 9
0: <
1: ≥
1 0 0
0 1
0 0 1
x
1 2 4
128 + 8
64 32 16
= 1+8+16 = 25

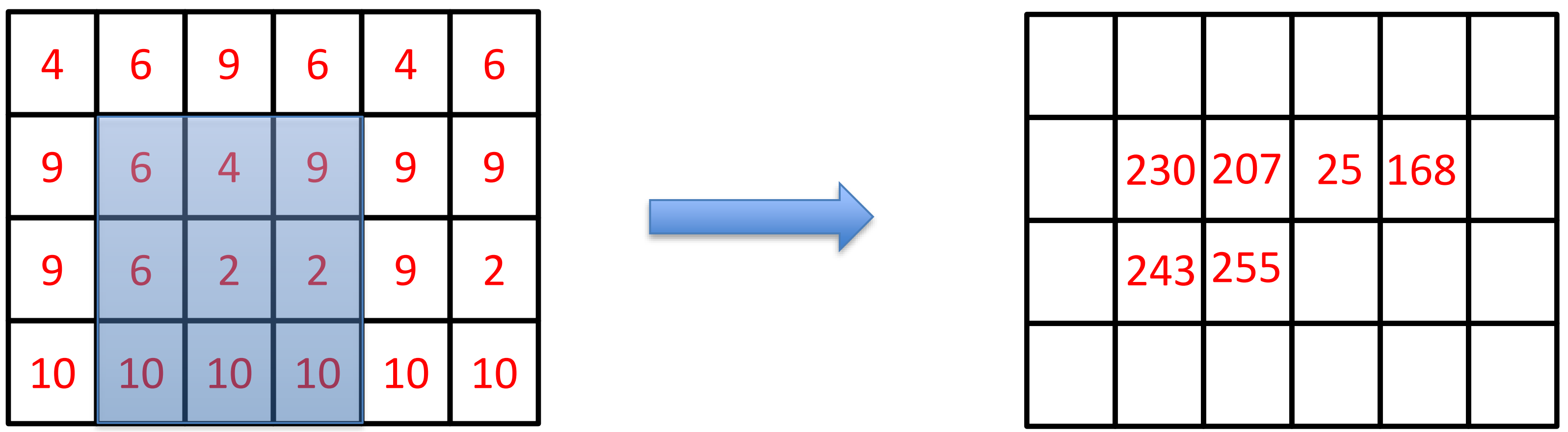
4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

	230	207	25	168	

4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

→

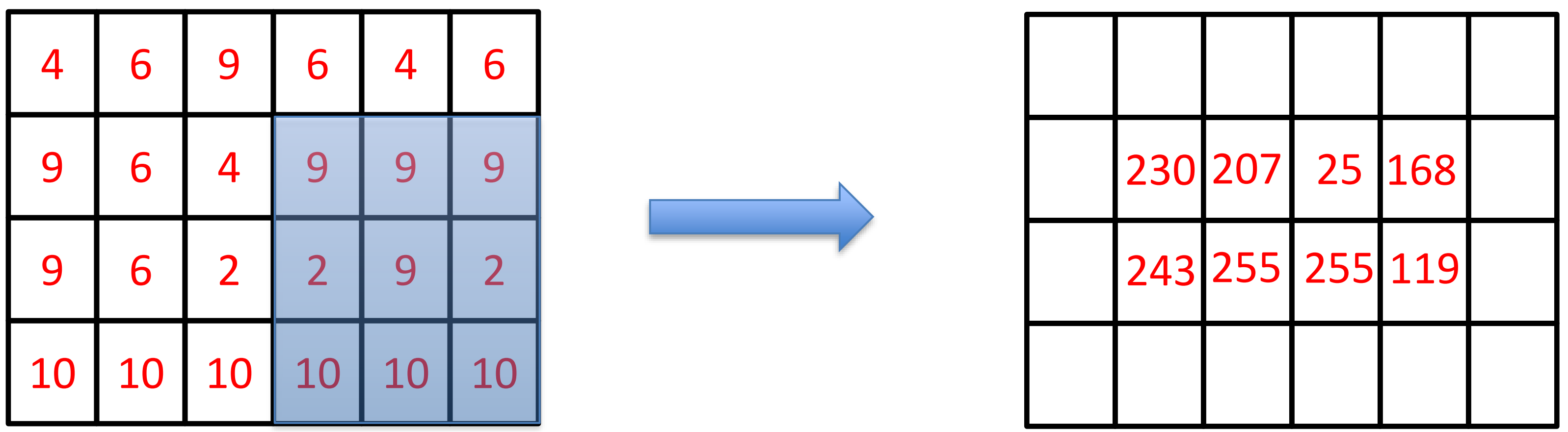
	230	207	25	168	
	243				



4 6 9 6 4 6
9 6 4 9 9 9
9 6 2 2 9 2
10 10 10 10 10 10
230 207 25 168
243 255

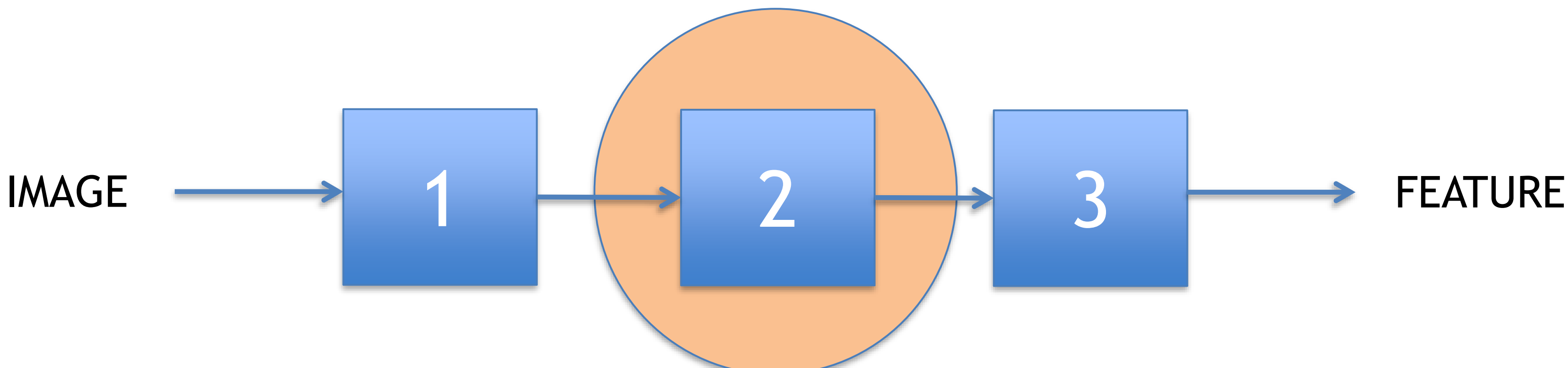
4	6	9	6	4	6
9	6	4	9	9	9
9	6	2	2	9	2
10	10	10	10	10	10

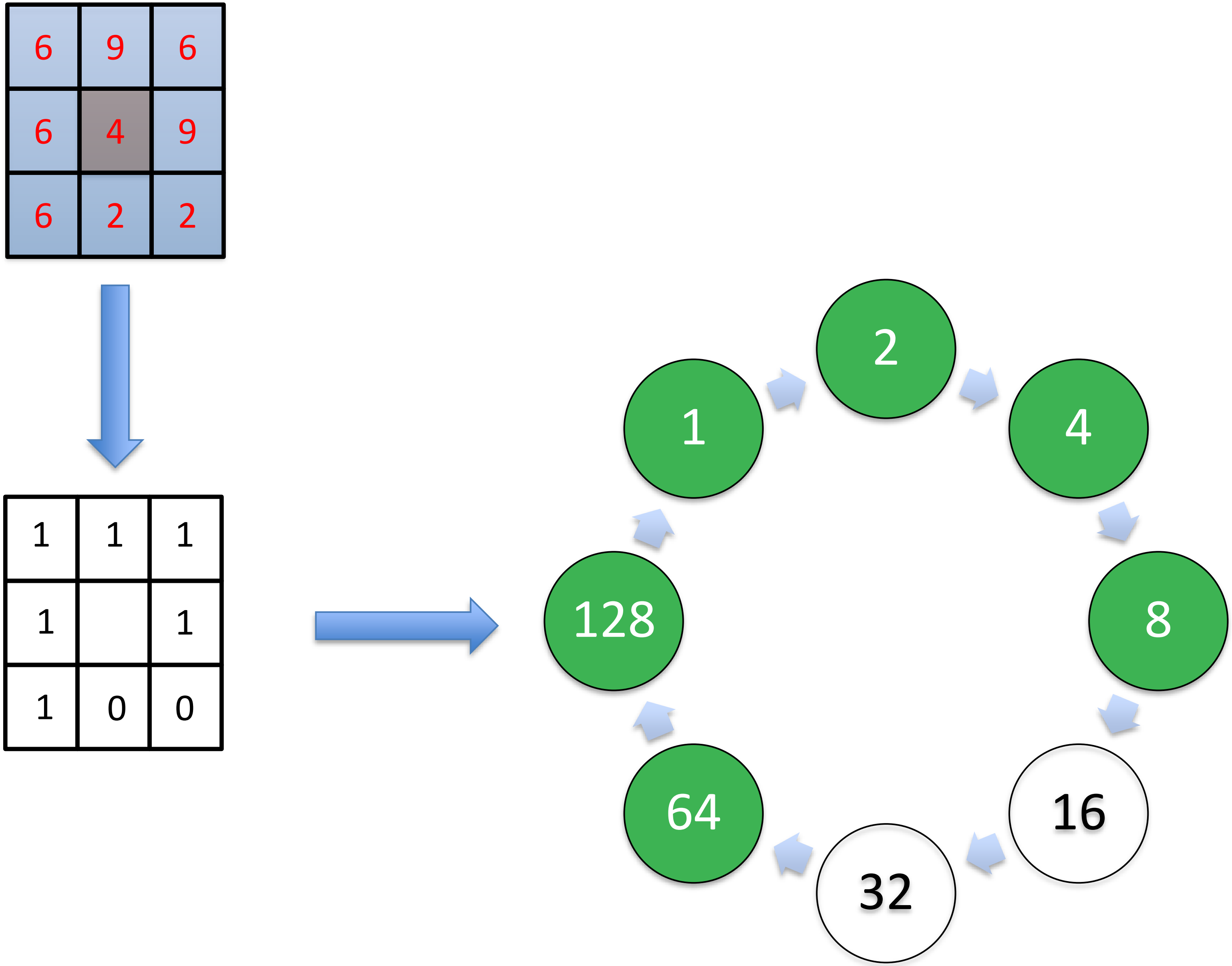
	230	207	25	168	
	243	255	255		


4 6 9 6 4 6
9 6 4 9 9 9
9 6 2 2 9 2
10 10 10 10 10 10
230 207 25 168
243 255 255 119

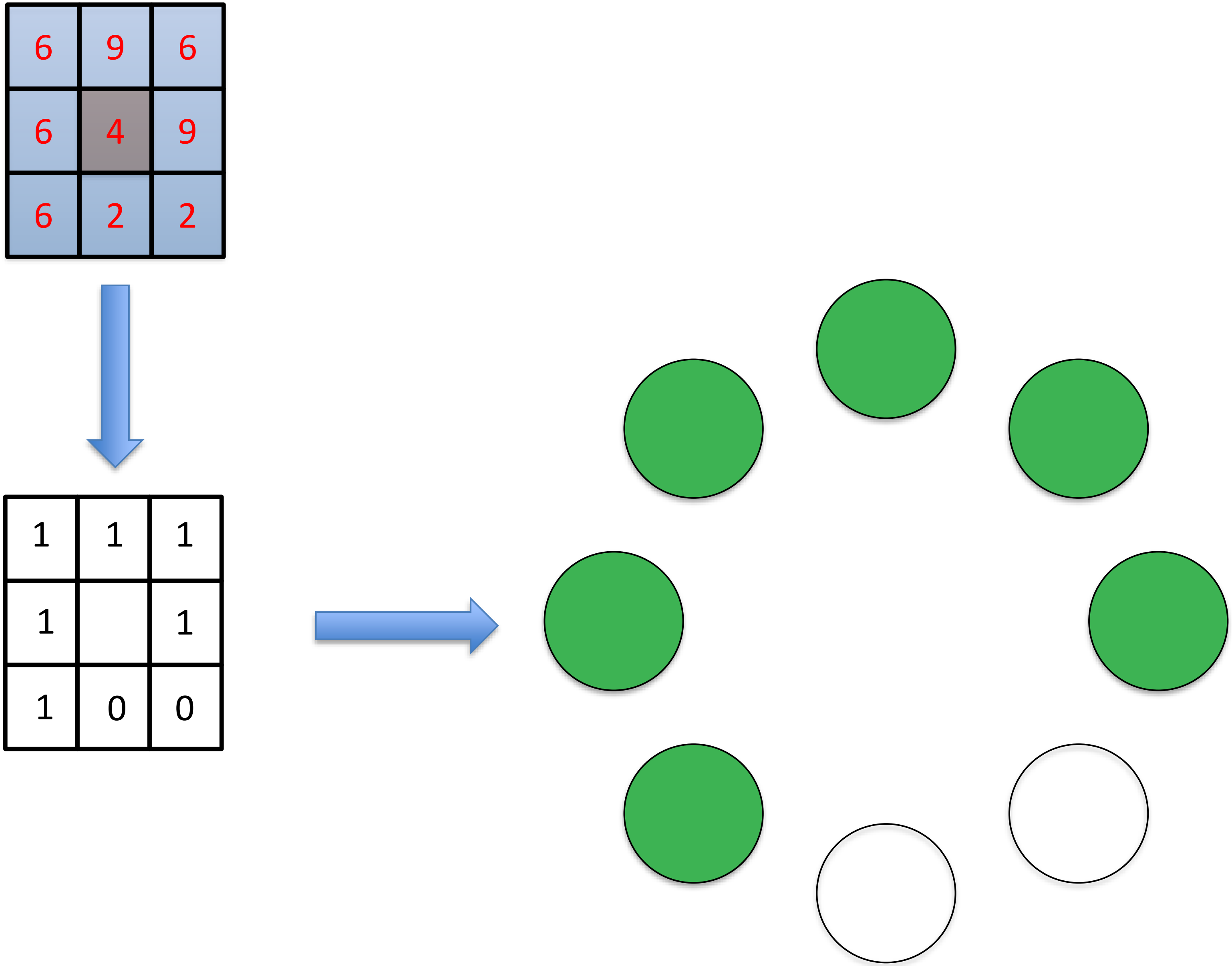
Local Binary Patterns

1. Coding
2. Mapping
3. Histogram



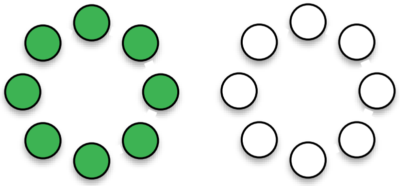


6
9
6
6
4
9
6
2
2
1
1
1
1
1
1
0
0
1
2
4
8
16
32
64
128

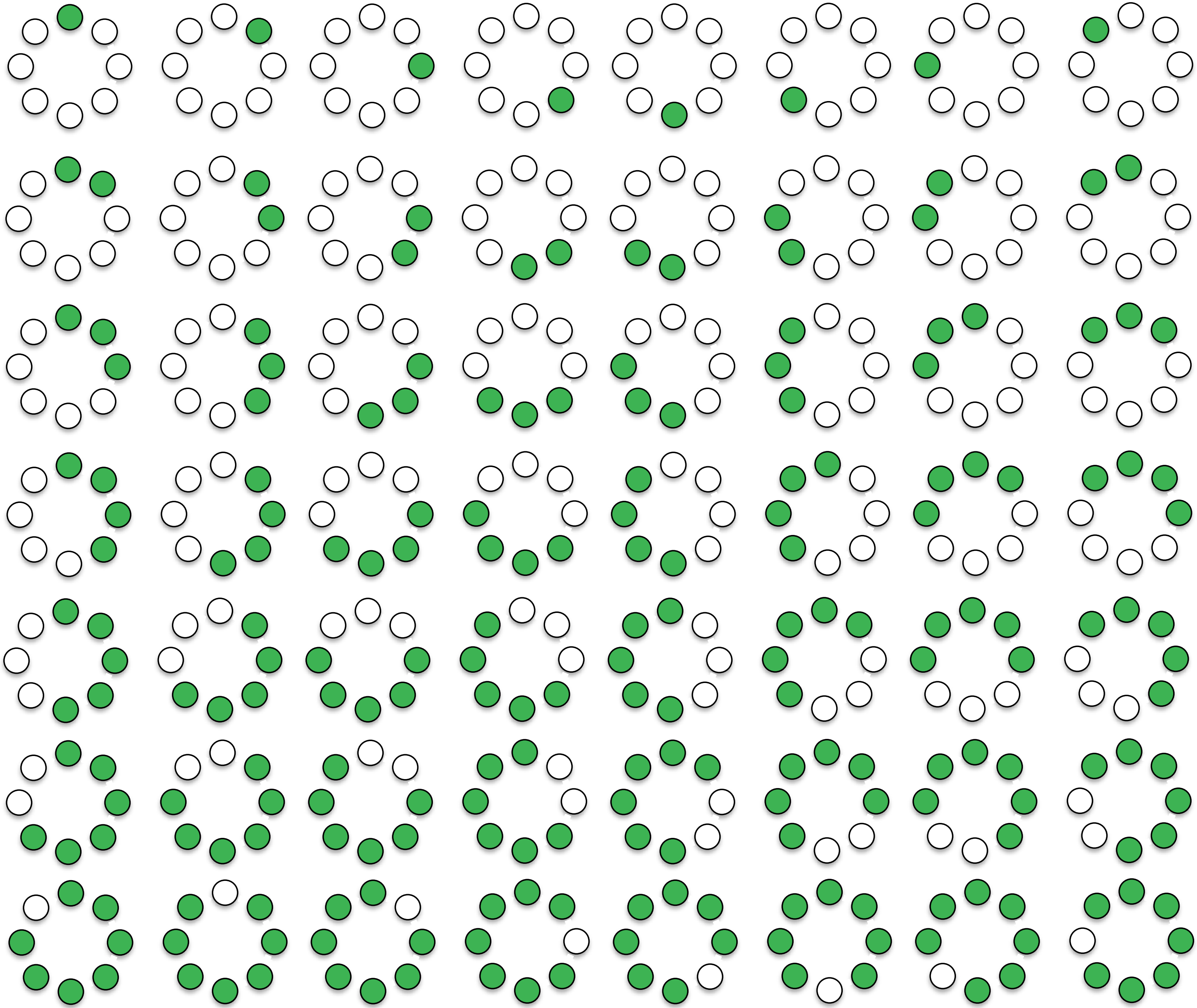

6 9 6
6 4 9
6 2 2
1 1 1
1 1
1 0 0

Uniform patterns

U = 0



U = 2



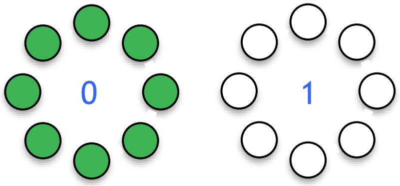
Uniform patterns

2 + 56 = 58 patterns

U = 0

2 patterns

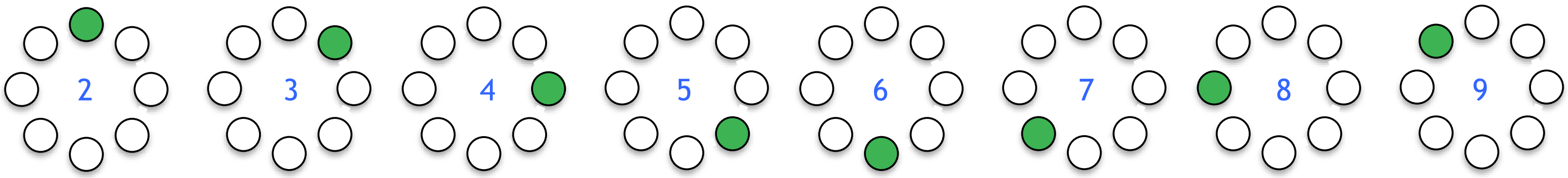
{0,1}



U = 2

8x7 = 56 patterns

{2, 3, ... 57}



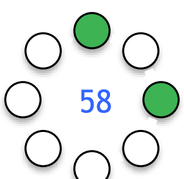
:

Non-uniform patterns

256 -58 = 198 patterns

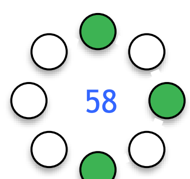
{58}

U = 4



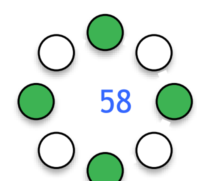
:

U = 6



:

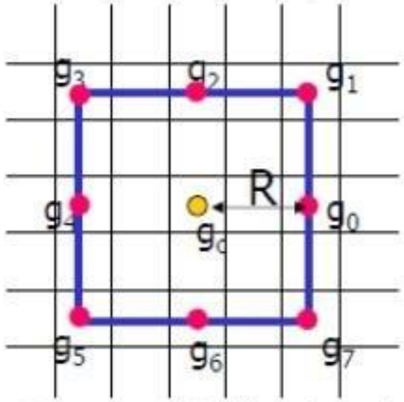
U = 8



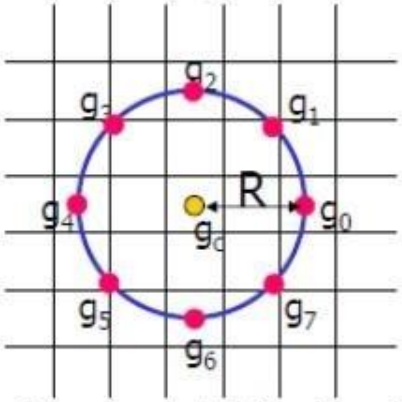
:

Description of Local image texture

- Texture at g_c is modeled using a local neighborhood of radius R, which is sampled at P (8 in the example) points:



Square neighborhood



Circular neighborhood
(g_1,g_3,g_5,g_7 interpolated)

- Let's define texture T as the joint distribution of the gray levels g_c and g_p ($p=0,\ldots,P-1$):

$$T = t(g_c , g_0 ,\ldots, g_{P-1})$$

Description of Local image texture (cont.)

- Without losing information, we can subtract g_c from g_p :

$$\mathbf{T = t(g_c\,, g_0\text{-}g_c\,,\ldots, g_{P\text{-}1}\text{-}g_c)}$$

- Assuming that g_c is independent of g_p-g_c , we can factorize above:

$$\mathbf{T \sim t(g_c)\; t(\,g_0\text{-}g_c\,,\ldots, g_{P\text{-}1}\text{-}g_c)}$$

- $t(g_c)$ describes the overall luminance of the image, which is unrelated to local image texture, hence we ignore it:

$$\mathbf{T \sim t(\,g_0\text{-}g_c\,,\ldots, g_{P\text{-}1}\text{-}g_c)}$$

Above expression is invariant wrt. Gray scale shifts

LBP: Local Binary Pattern

- Invariance wrt. to any monotonic transformation of the gray scale is achieved by considering the signs of the difference:

$$T \sim t(\, s(g_0 - g_c), \ldots, s(g_{P-1} - g_c))$$

Where

$$s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

- Above is transformed into a unique P-bit pattern code by assigning binomial coefficient 2^p to each $s(g_p - g_c)$:

$$LBP_{P,R} = \sum_{p=0}^{P-1} s\,(g_p - g_c)\, 2^p$$

LBP: Local Binary Pattern (cont.)

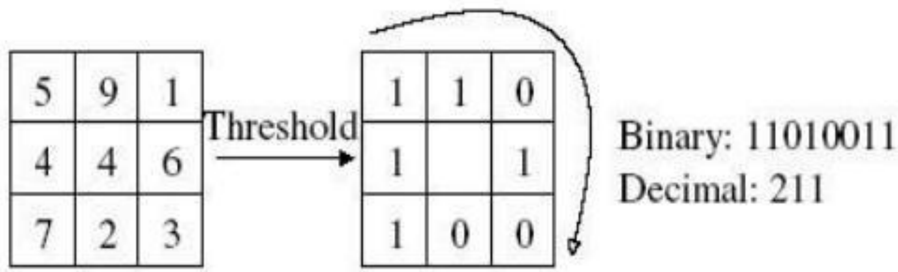


Fig. 1. The basic LBP operator.

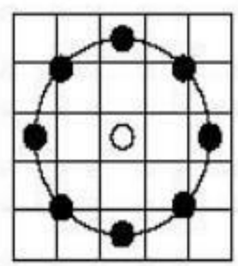
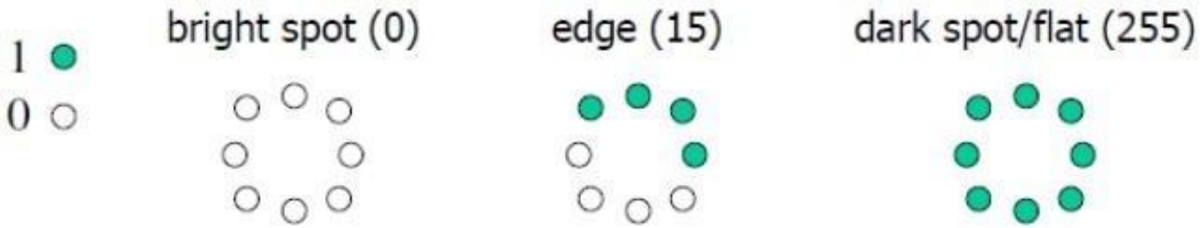


Fig. 2. The circular (8,2) neigbourhood. The pixel values are bilinearly interpolated whenever the sampling point is not in the center of a pixel.

LBP: Local Binary Pattern (cont.)

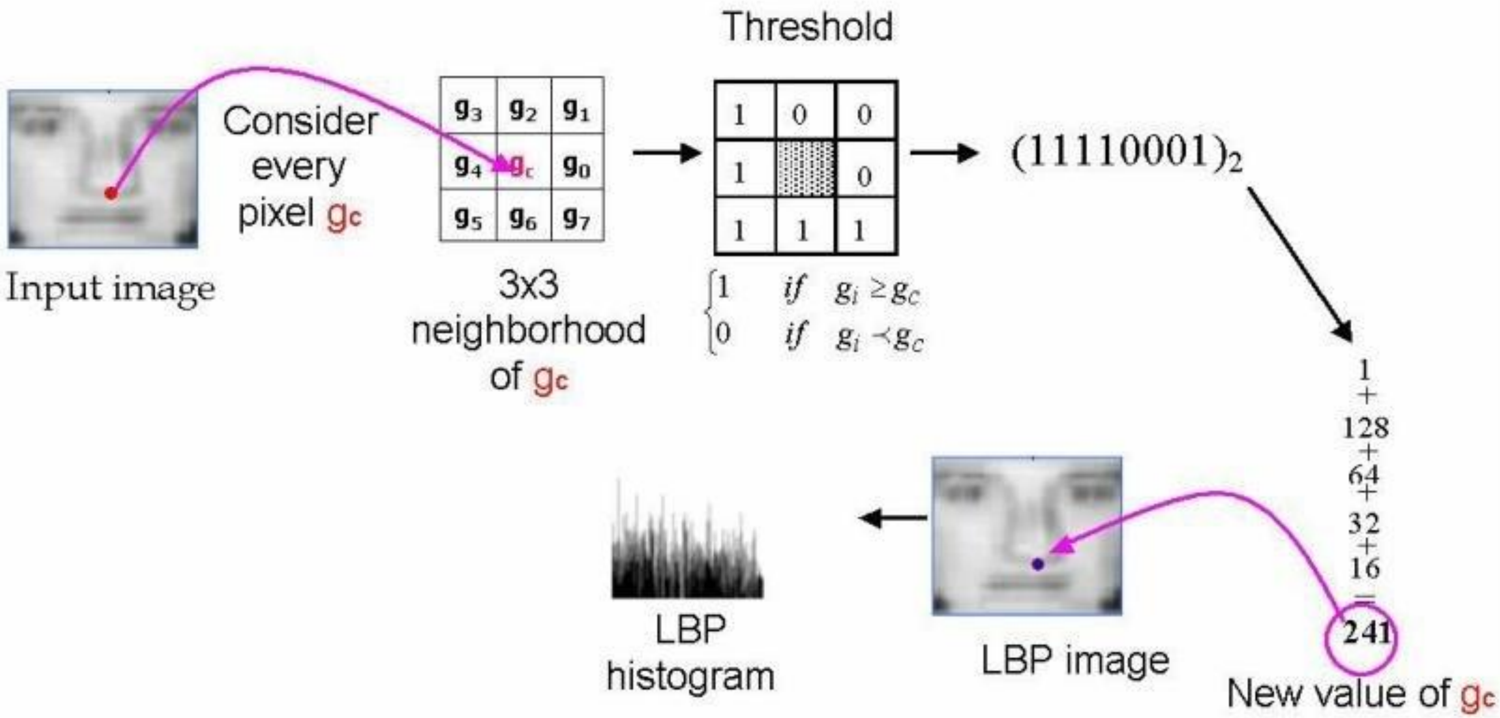
- $LBP_{P,R}$ encodes simple binary microstructures into P-bit number:



- $LBP_{P,R}$ provides less information than signed difference p_8 but:
  - invariant wrt. To any monotonic transformation of the gray scale
  - Vector quantization not needed
  - Computational simplicity

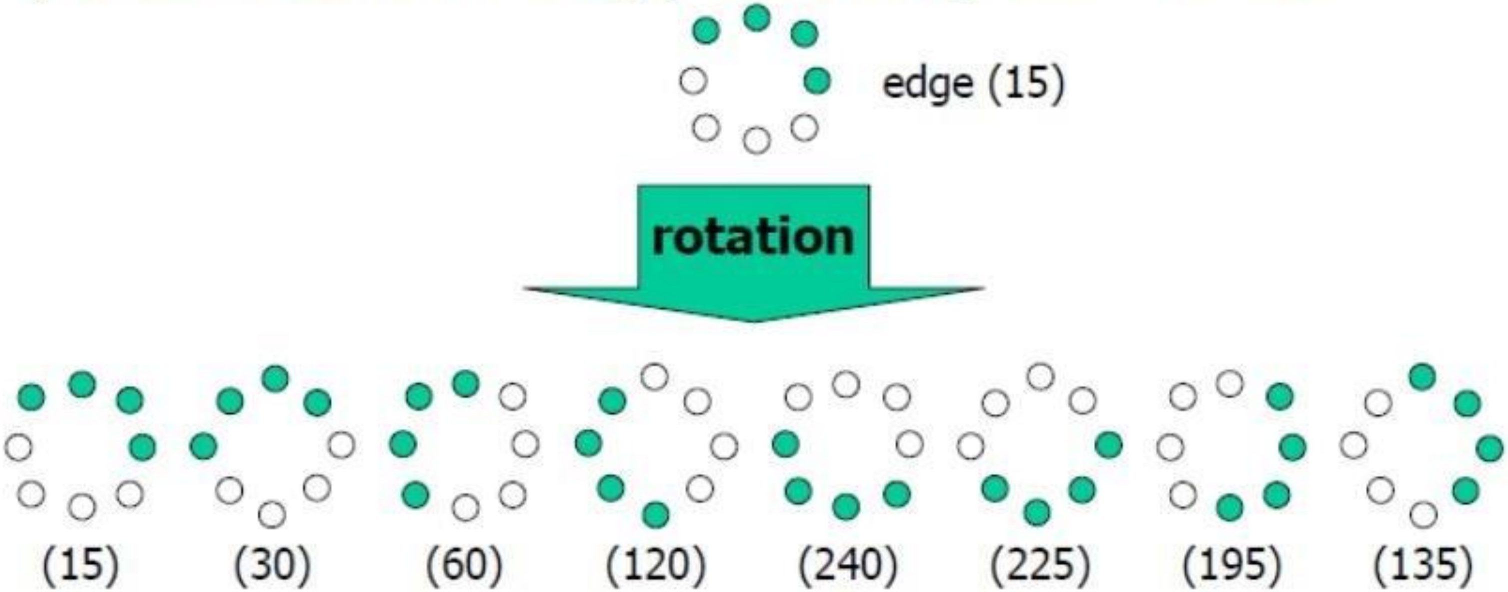
LBP: Example

- Local Binary Pattern (LBP) is a texture descriptor which codifies local primitives (such as curved edges, spots, flat areas, etc.) into a feature histogram.



Rotation invariant LBP

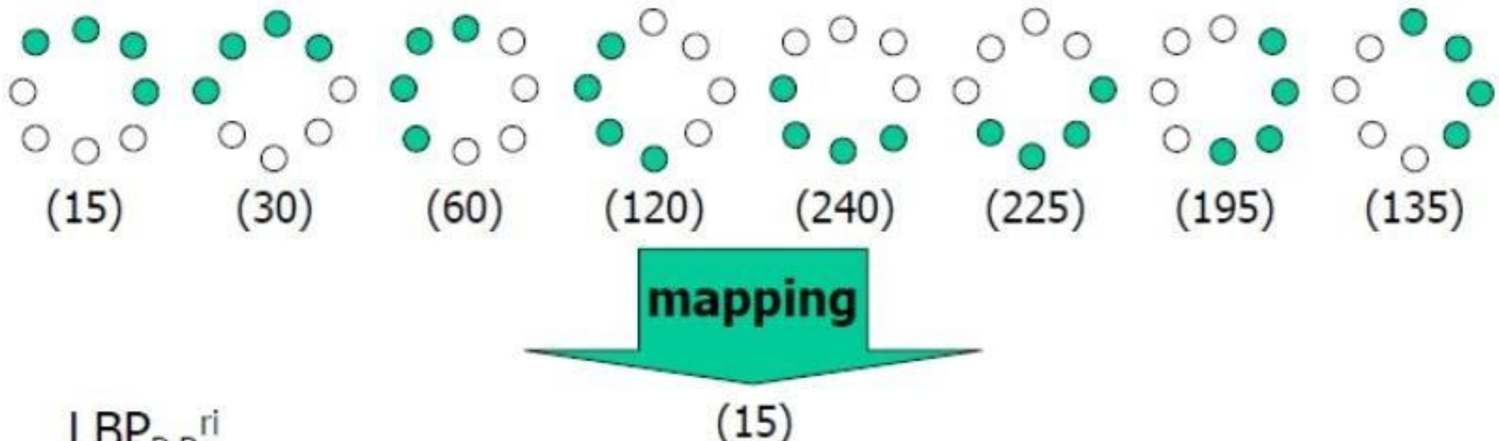
Spatial rotation of the binary pattern changes the LBP code:



Rotation invariant LBP (cont.)

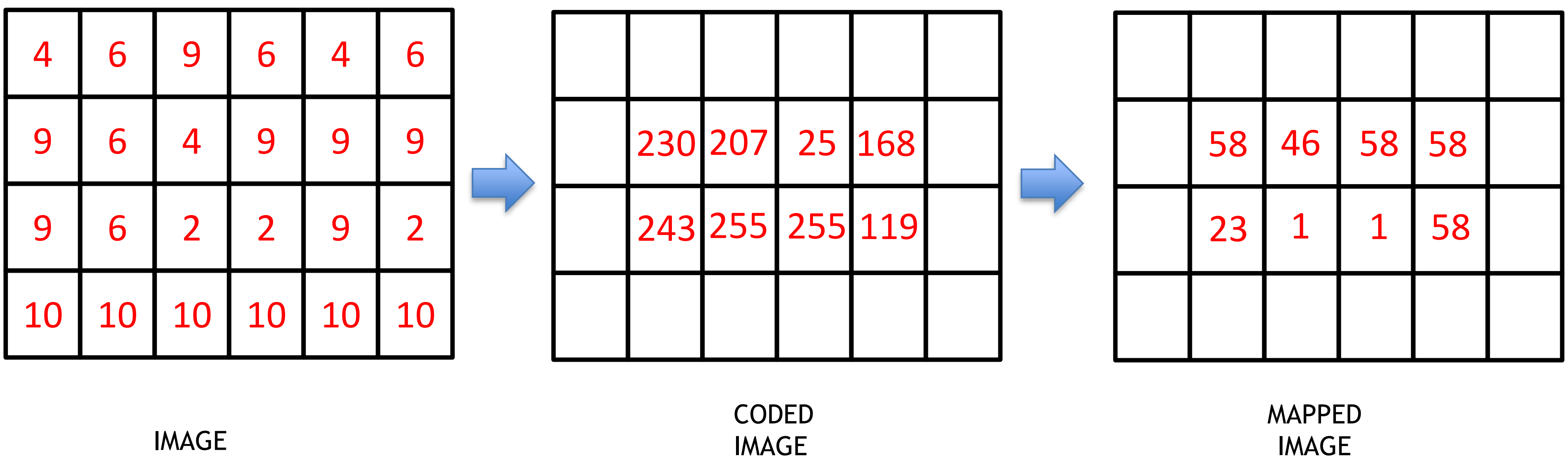
Formally, rotation invariance can be achieved by defining:

$$LBP_{P,R}^{ri} = \min\{ROR(LBP_{P,R}, i) \mid i=0, \ldots, P-1\}$$



$LBP_{P,R}^{ri}$

- invariant wrt. any monotonic gray scale transformation
- invariant wrt. rotation along the circular neighborhood


4 6 9 6 4 6
9 6 4 9 9 9
9 6 2 2 9 2
10 10 10 10 10 10
IMAGE
230 207 25 168
243 255 255 119
CODED IMAGE
58 46 58 58
23 1 1 58
MAPPED IMAGE

Uniform Pattern

<u>Heuristic hypothesis</u>

- Certain local binary patterns are fundamental properties of texture, providing a vast majority, sometimes overall 90%, of all 3x3 patterns in the observed textures:
  - Define the concept of 'uniform' patterns, which have a limited number of spatial transitions
  - Use only uniform patterns
  - Exclude 'nonuniform' patterns of high angular frequency (they provide statistically unreliable information)

Uniform Pattern (cont.)

Pattern 'uniformity' measure $U(LBP_{P,R}^{ri})$:

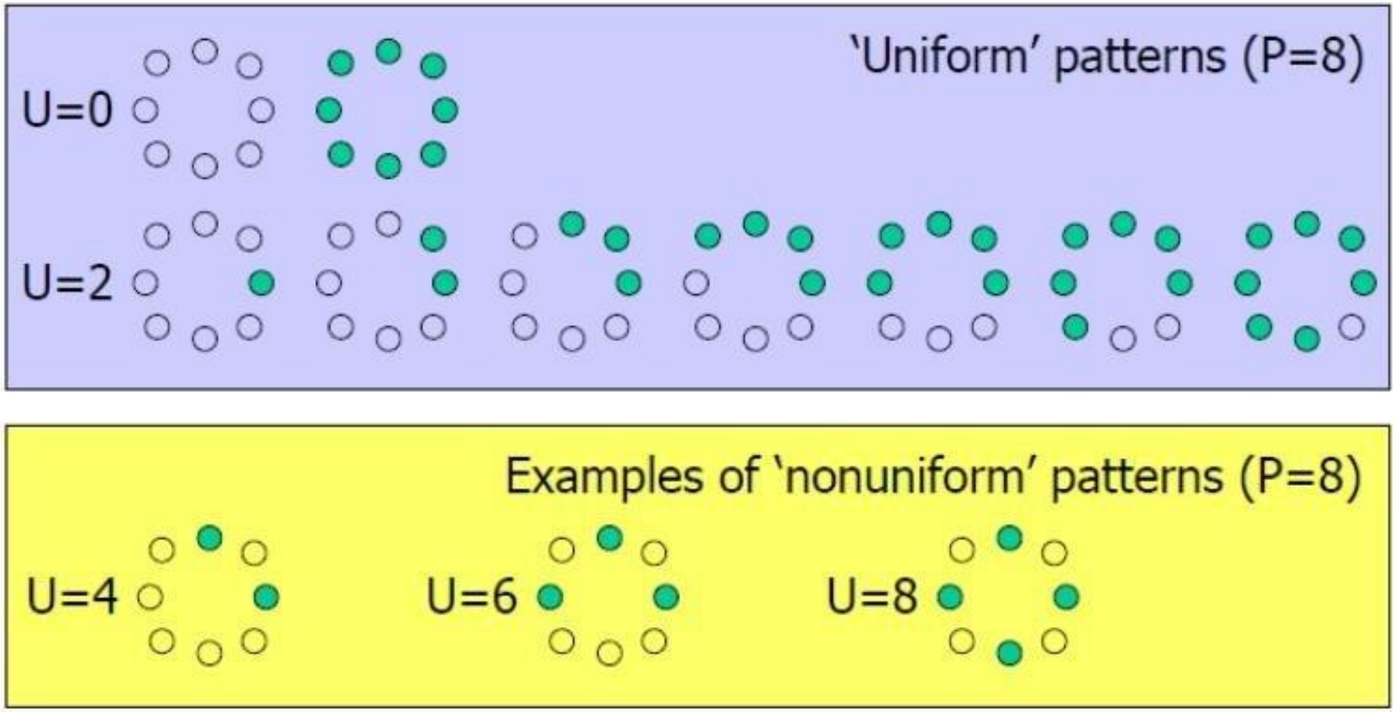
$$U(LBP_{P,R}) = |s(g_{P-1}-g_c) - s(g_0-g_c)| + \sum_{p=1}^{P-1} |s(g_p-g_c) - s(g_{p-1}-g_c)|$$

$U(LBP_{P,R})$ ~ # bitwise 0/1 transitions in the circular bit pattern

Pattern is considered 'uniform', if $U(LBP_{P,R}) <= 2$

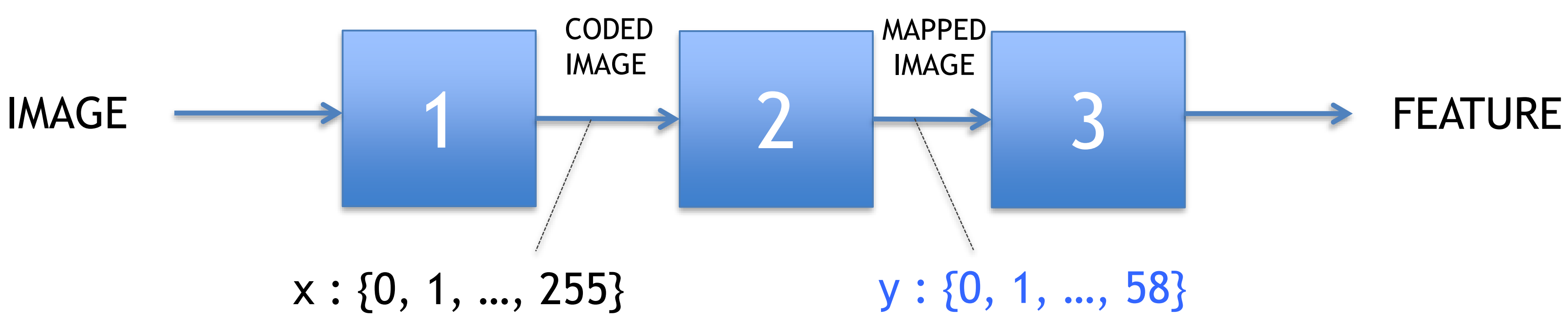
i.e. pattern has at most 1 bitwise spatial transition

Uniform Pattern (cont.)



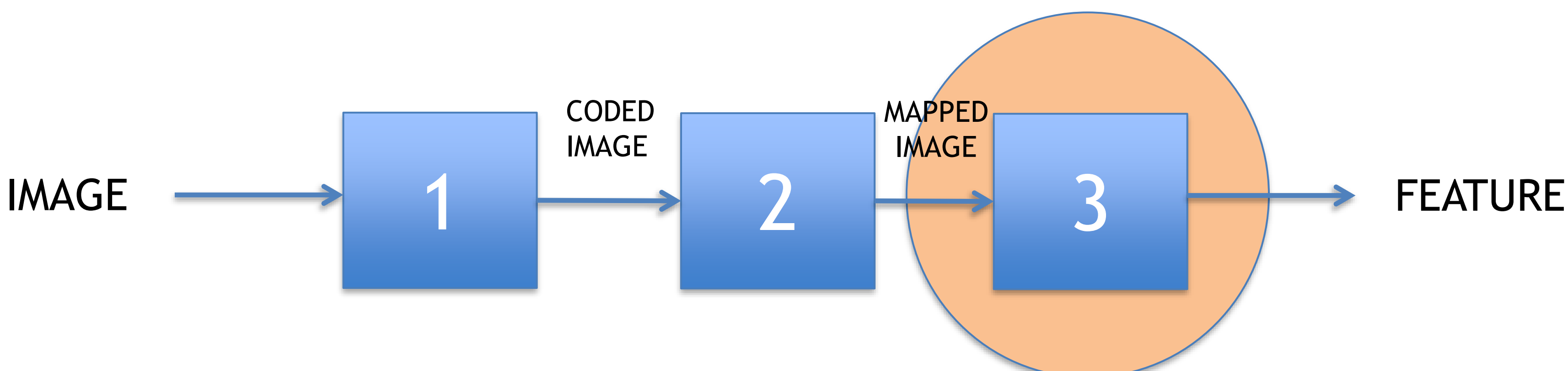
Local Binary Patterns

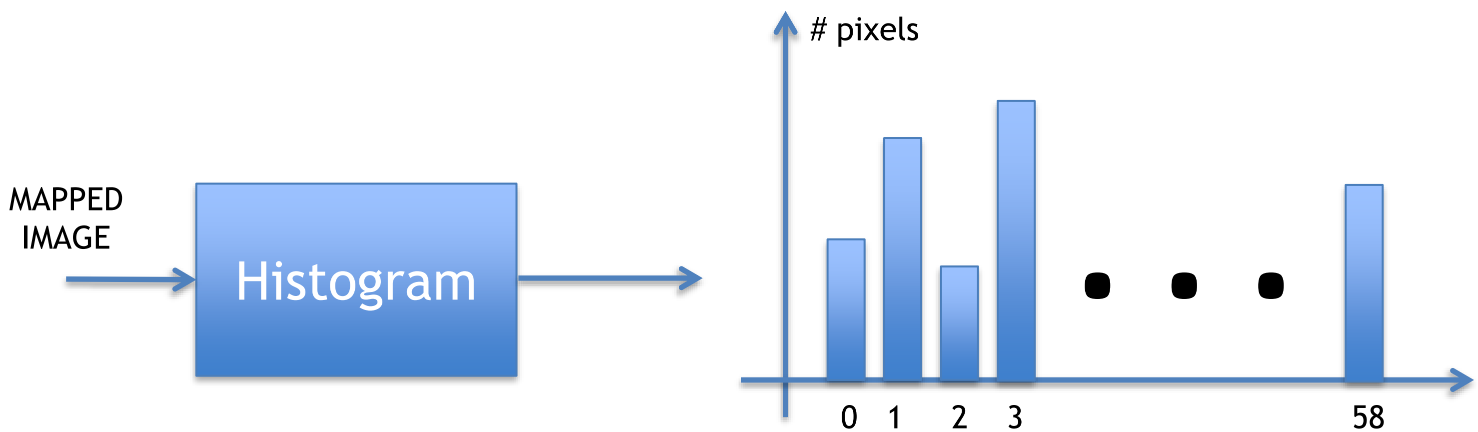
1. Coding
2. **Mapping**
3. Histogram



Local Binary Patterns

1. Coding
2. Mapping
3. Histogram

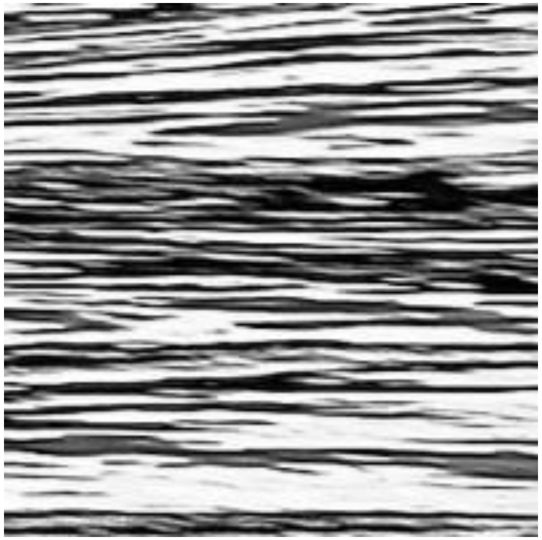


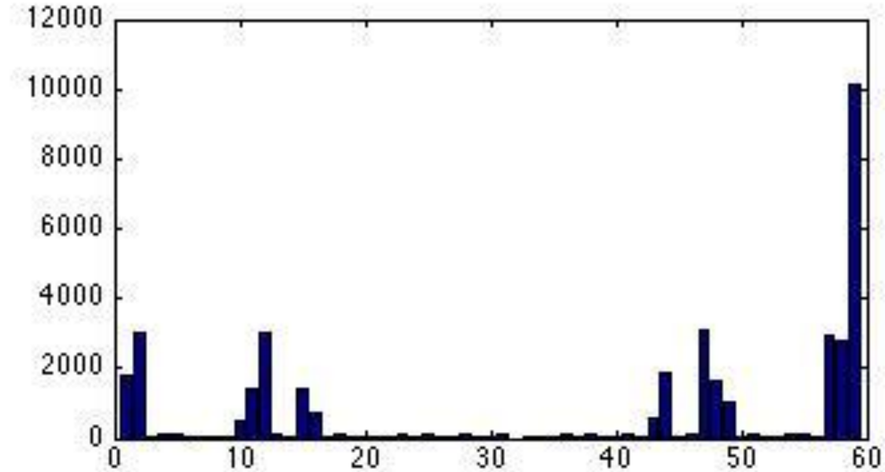


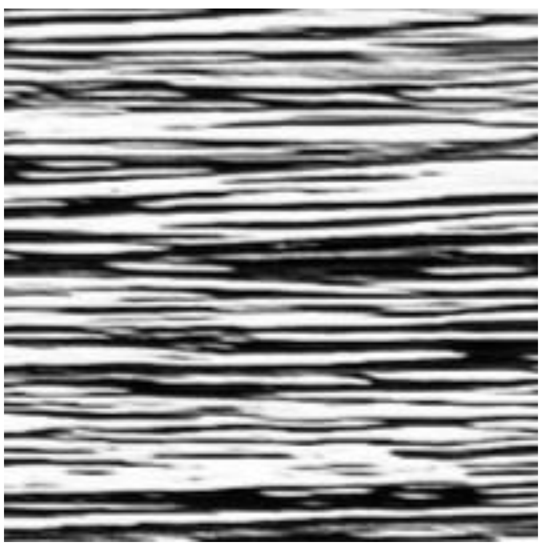
The image is described as a vector of 59 elements. Similar images have similar LBP features!!!

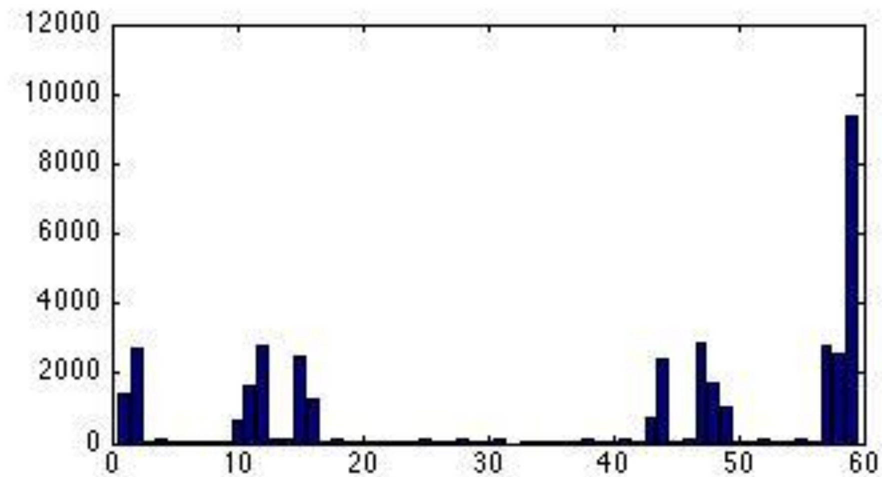
Local Binary Patterns

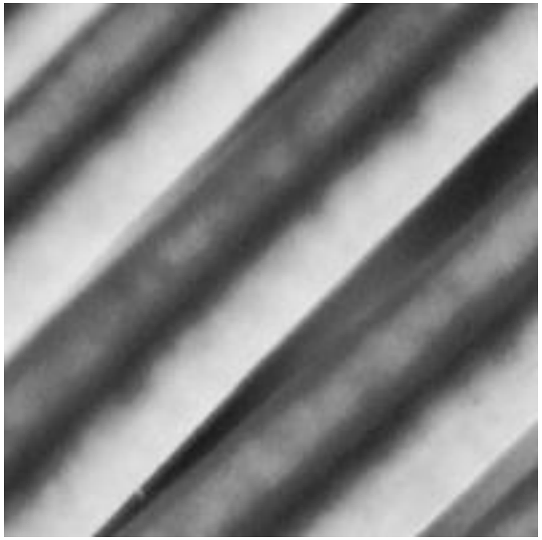
Examples
Texture Images

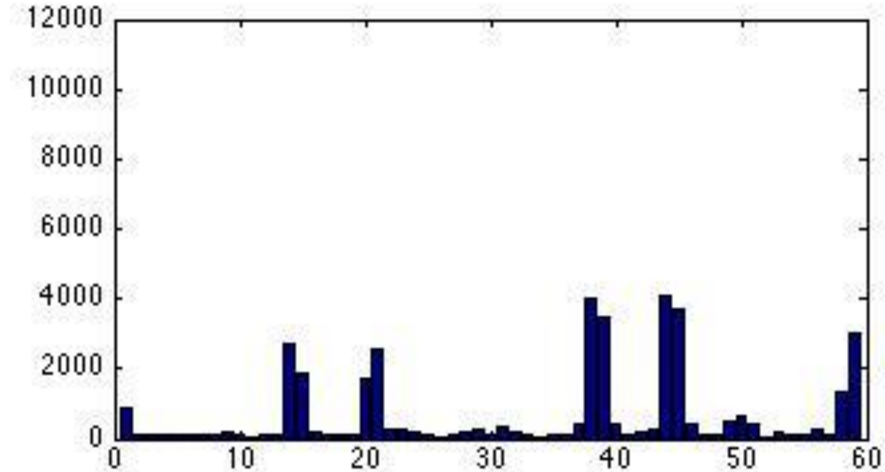


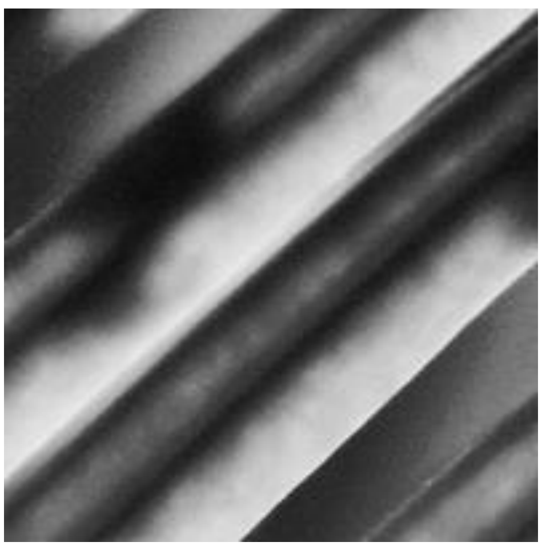

12000
10000
8000
6000
4000
2000
0
0
10
20
30
40
50
60

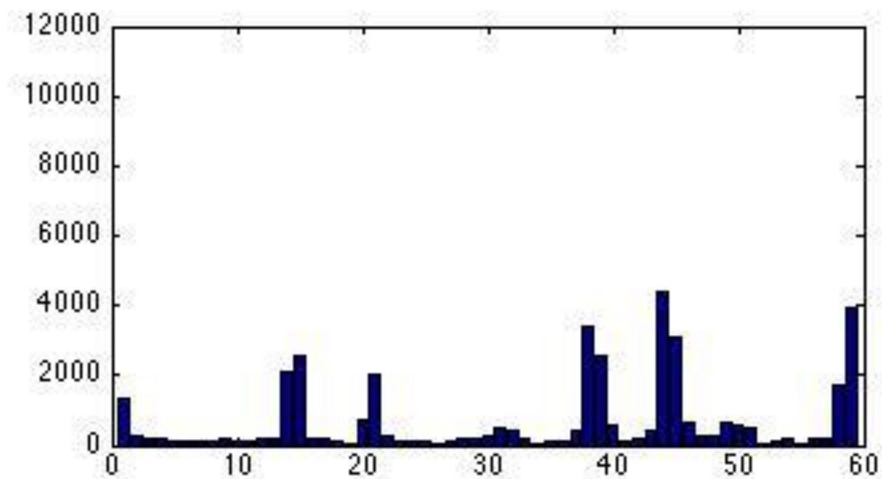


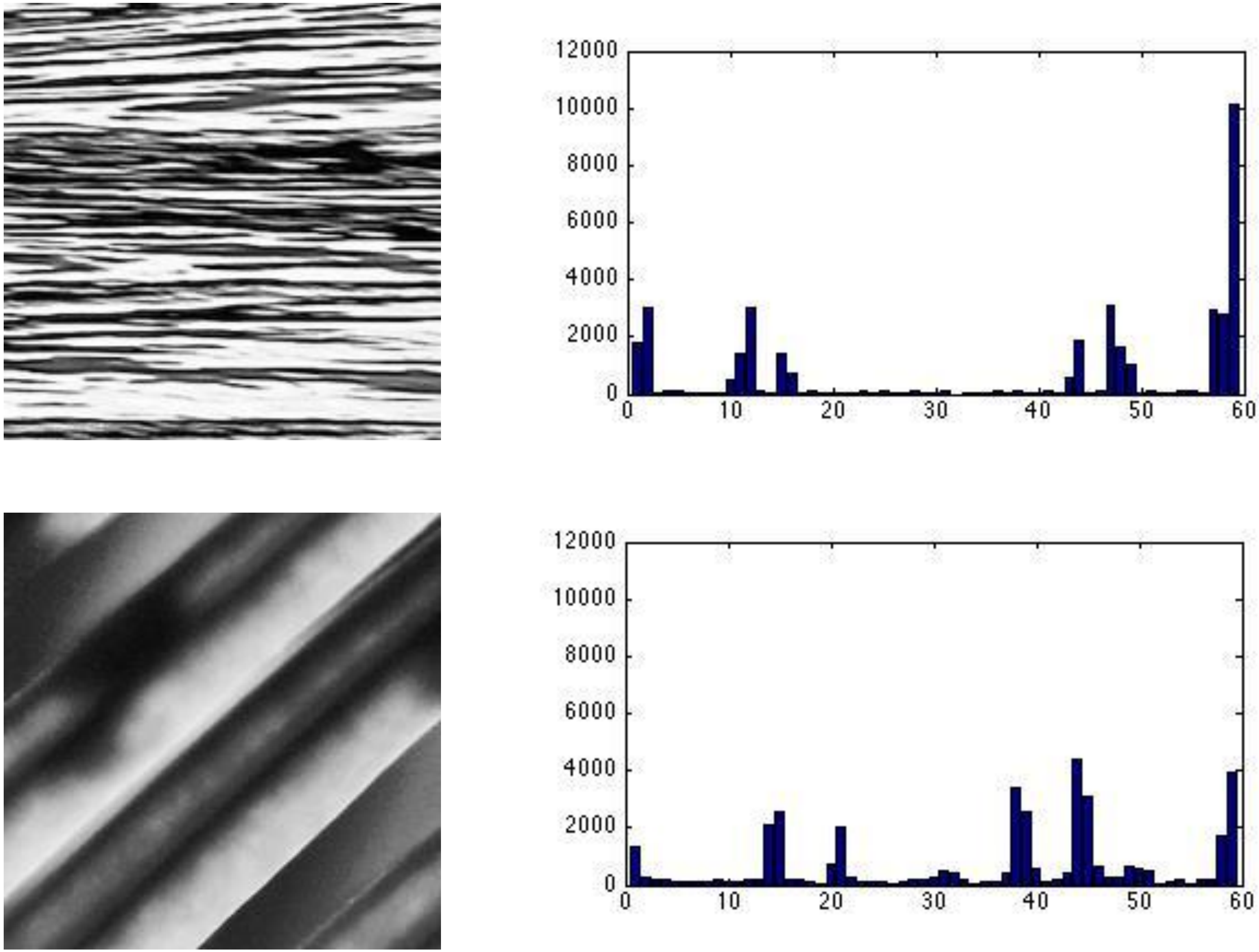

12000
10000
8000
6000
4000
2000
0
0
10
20
30
40
50
60

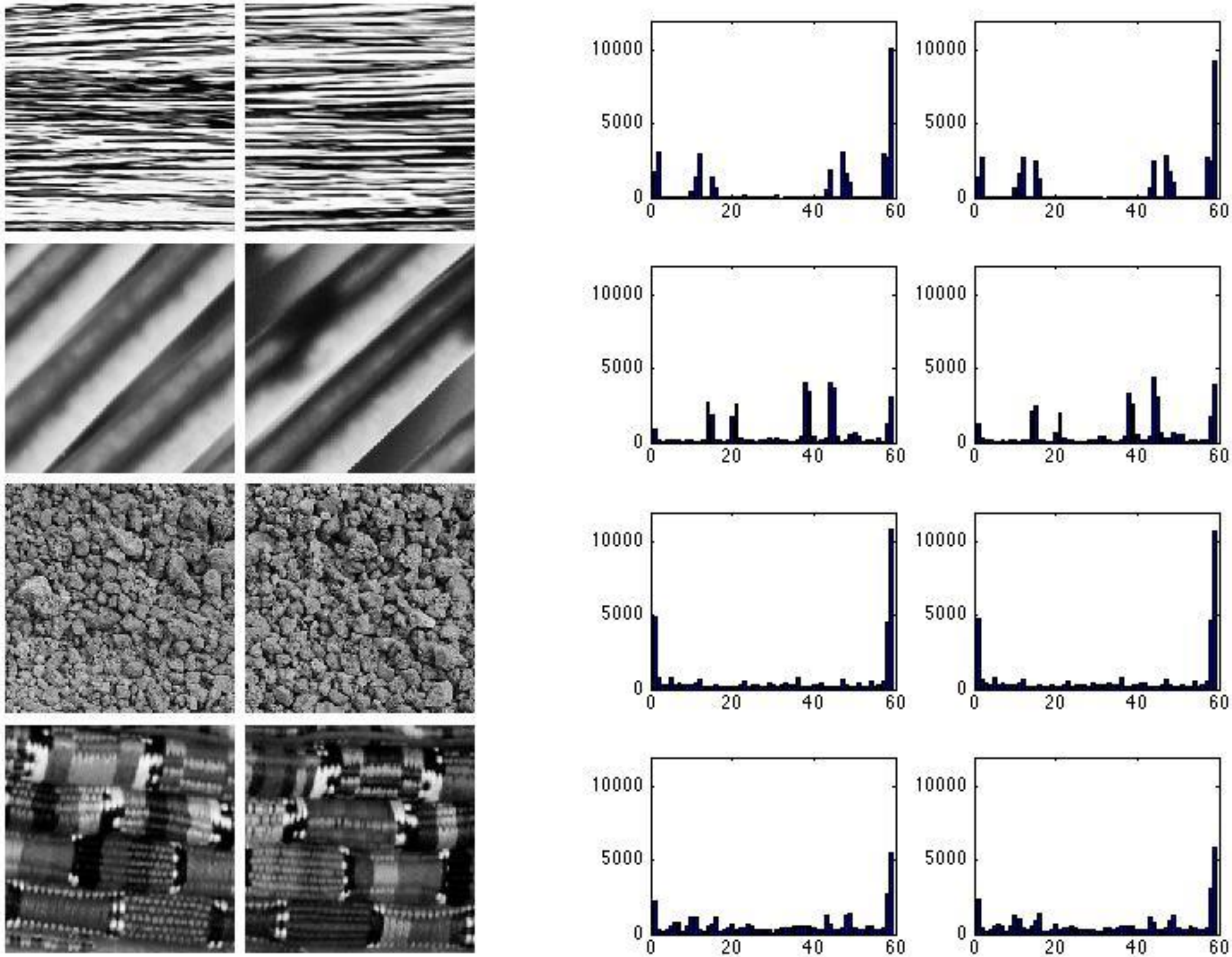



12000
10000
8000
6000
4000
2000
0
0
10
20
30
40
50
60

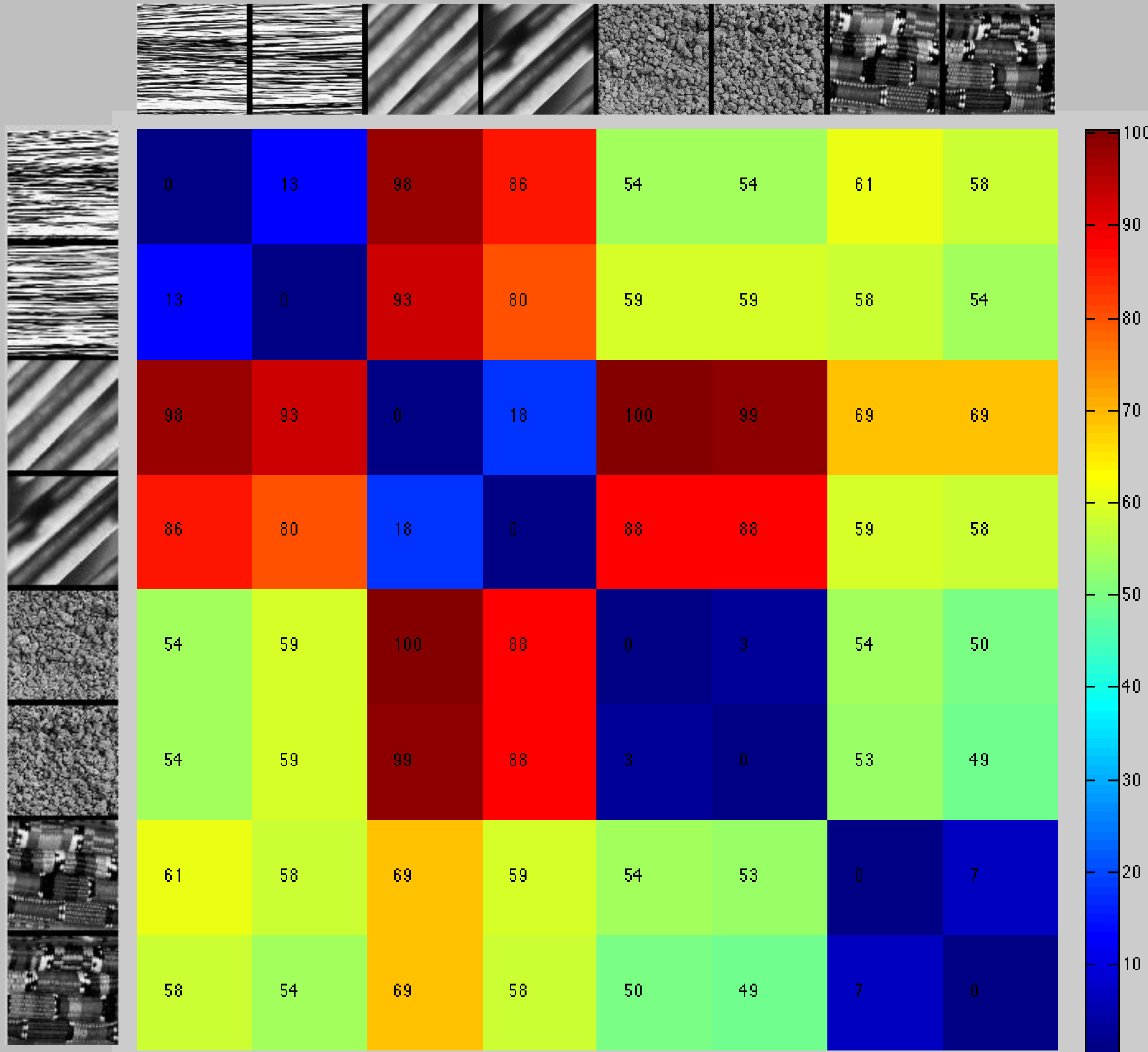



12000
10000
8000
6000
4000
2000
0
0
10
20
30
40
50
60


12000
10000
8000
6000
4000
2000
0
0 10 20 30 40 50 60
12000
10000
8000
6000
4000
2000
0
0 10 20 30 40 50 60



10000
5000
0
0
20
40
60


0 13 98 86 54 54 61 58
13 0 93 80 59 59 58 54
98 93 0 18 100 99 69 69
86 80 18 0 88 88 59 58
54 59 100 88 0 3 54 50
54 59 99 88 3 0 53 49
61 58 69 59 54 53 0 7
58 54 69 58 50 49 7 0
100
90
80
70
60
50
40
30
20
10

Local Binary Patterns

Examples
Face Recognition

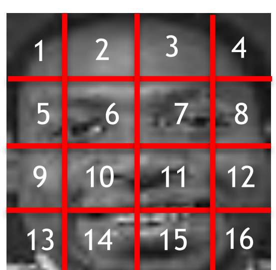
In the training set there are *k* classes.

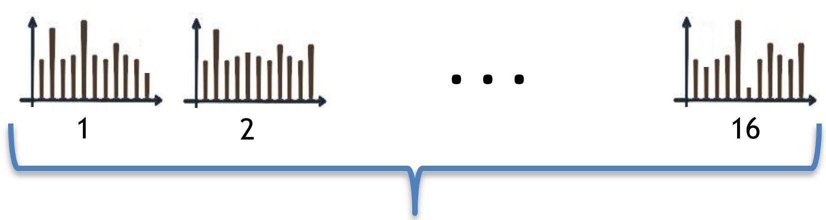
For each class we have *n* training images.

In this example there are 40 classes with 9 images each.

Each image we use w x *w* partitions

In each partition we extract LBP feature





A face is described using a feature of 16 x 59 = 944 elements

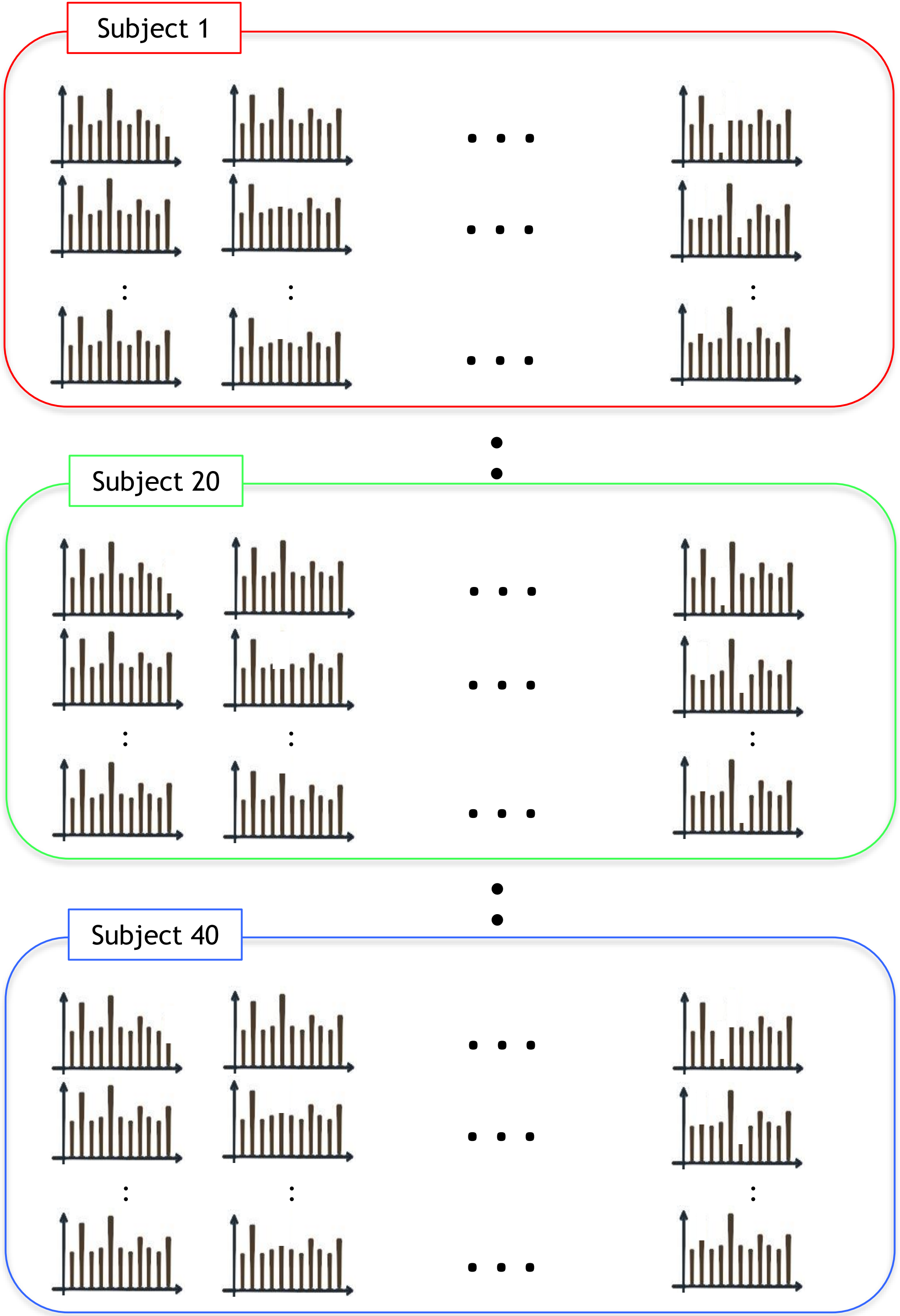
TRAINING

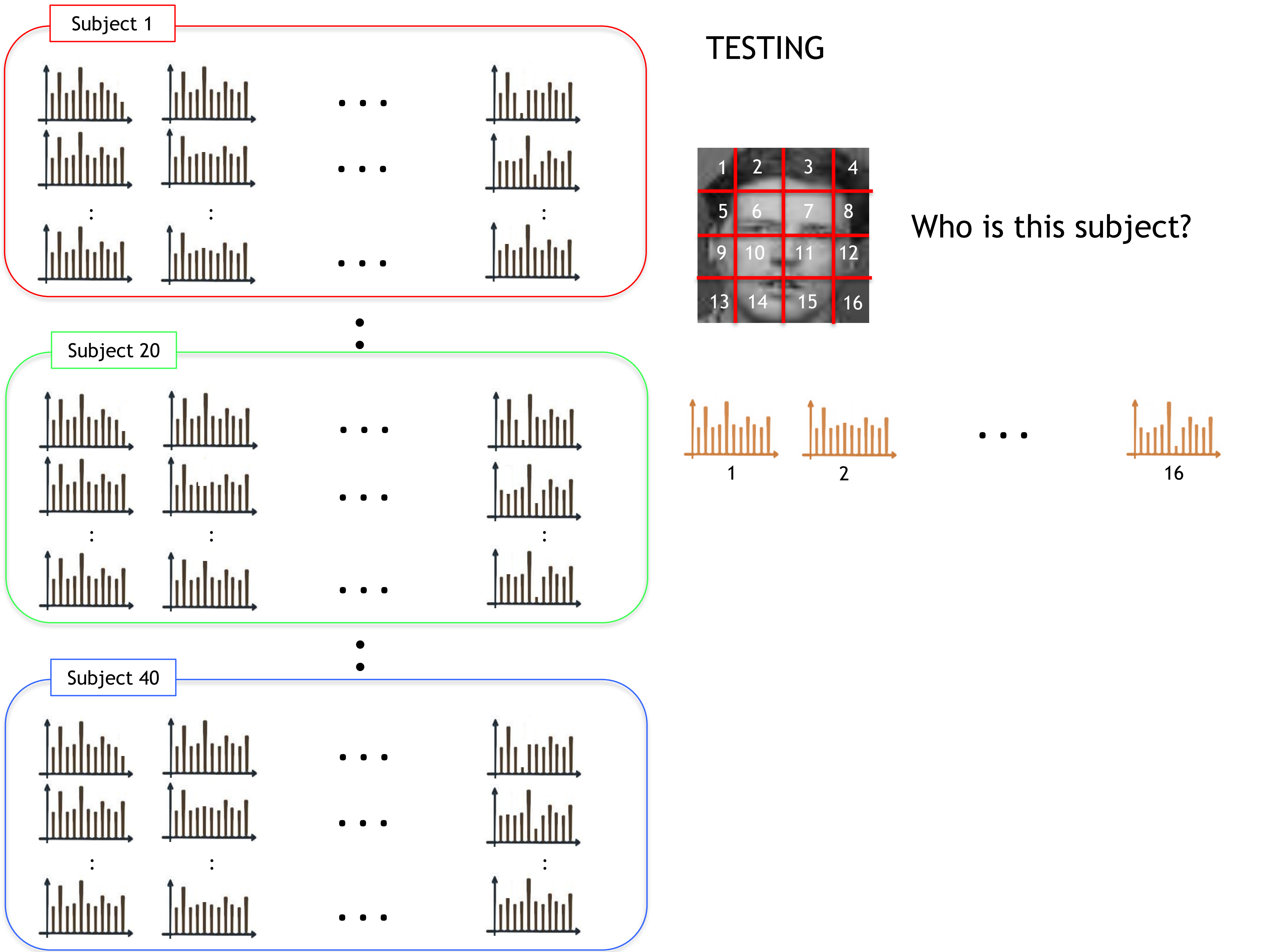
Training Data:

Table with:

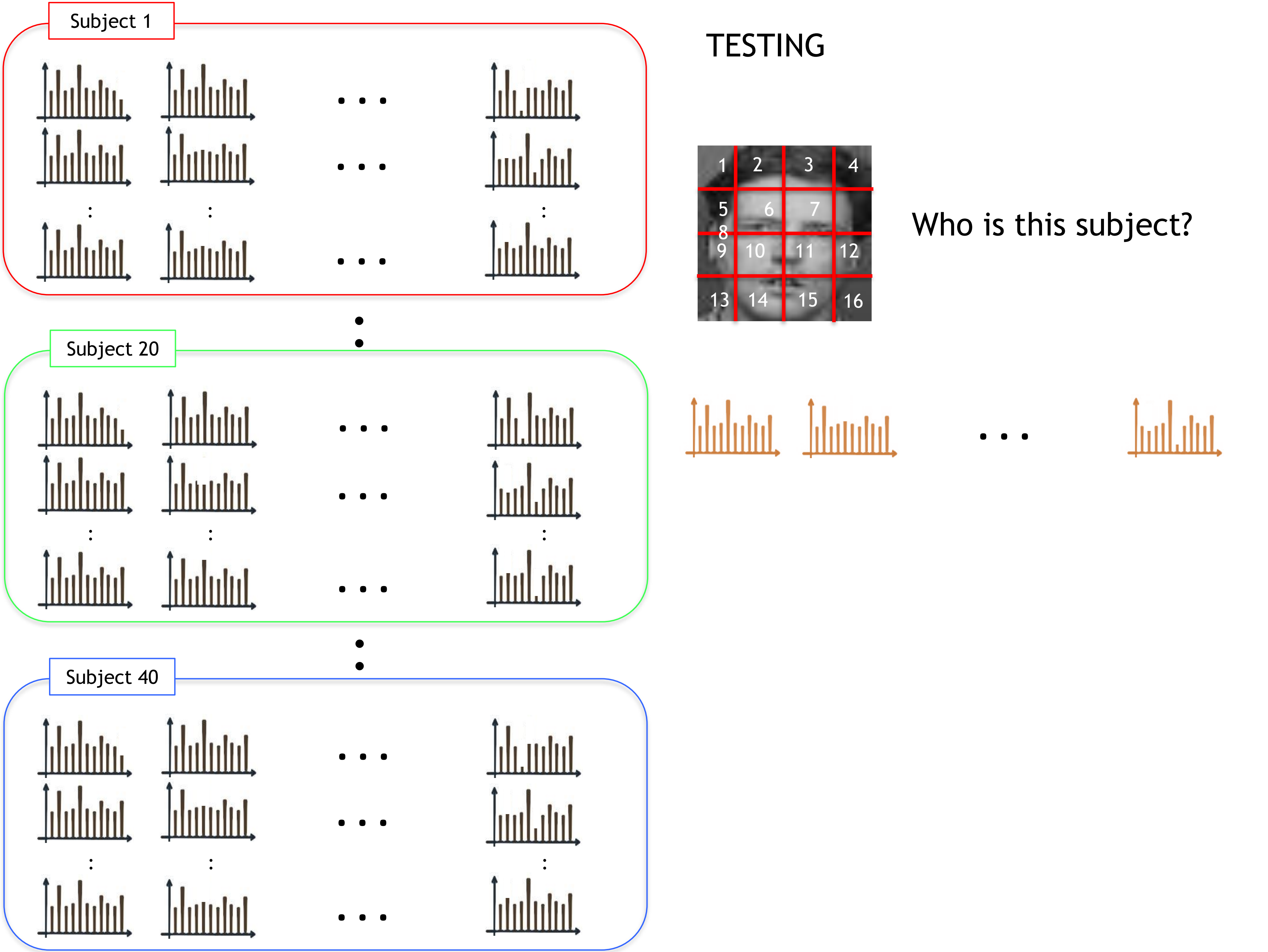
9x40 = 360 rows

16 x 59 = 944 columns

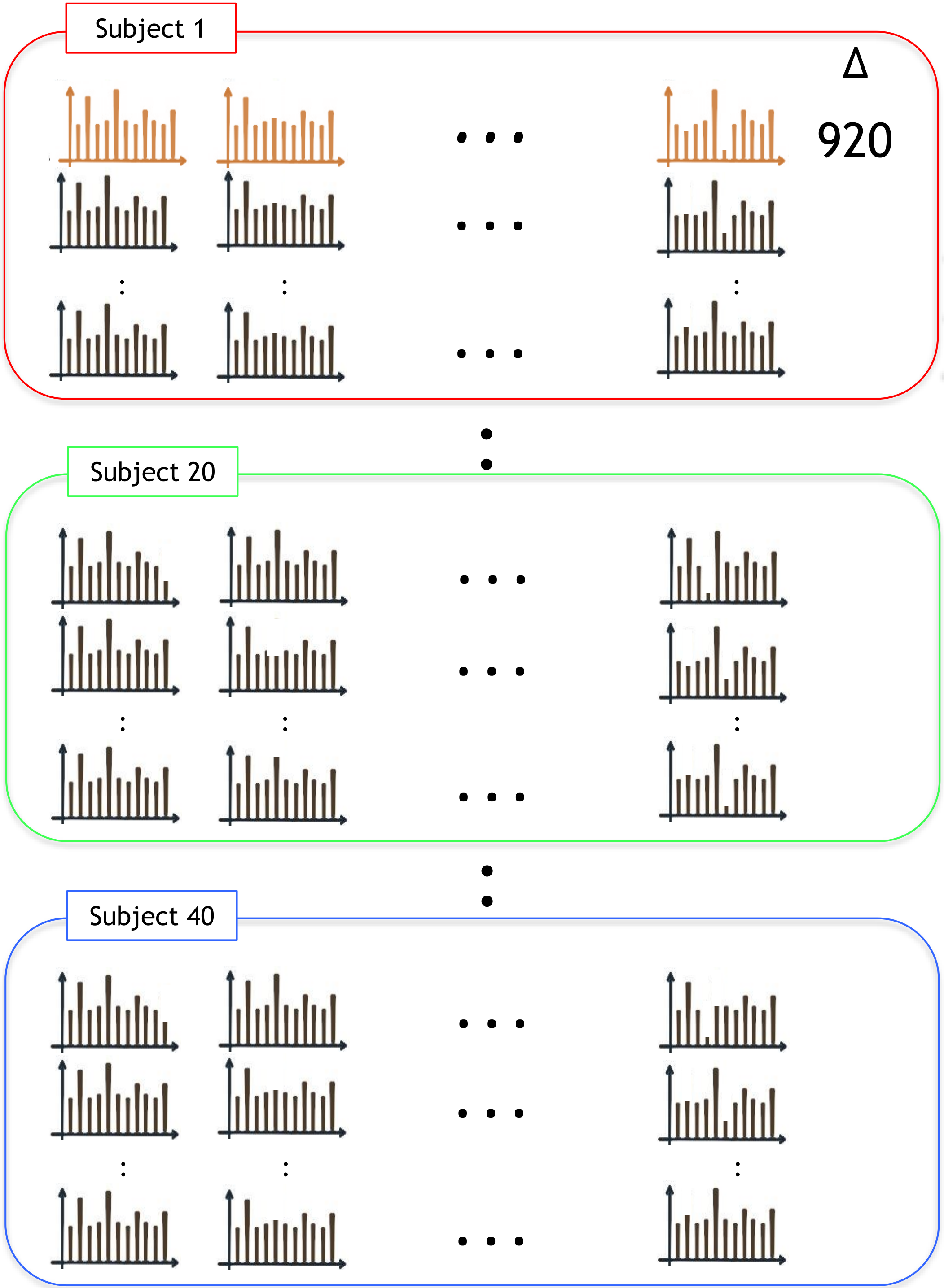

Subject 1
Subject 20
Subject 40



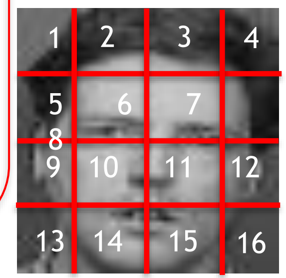
Subject 1
Subject 20
Subject 40
TESTING
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
Who is this subject?
1
2
16



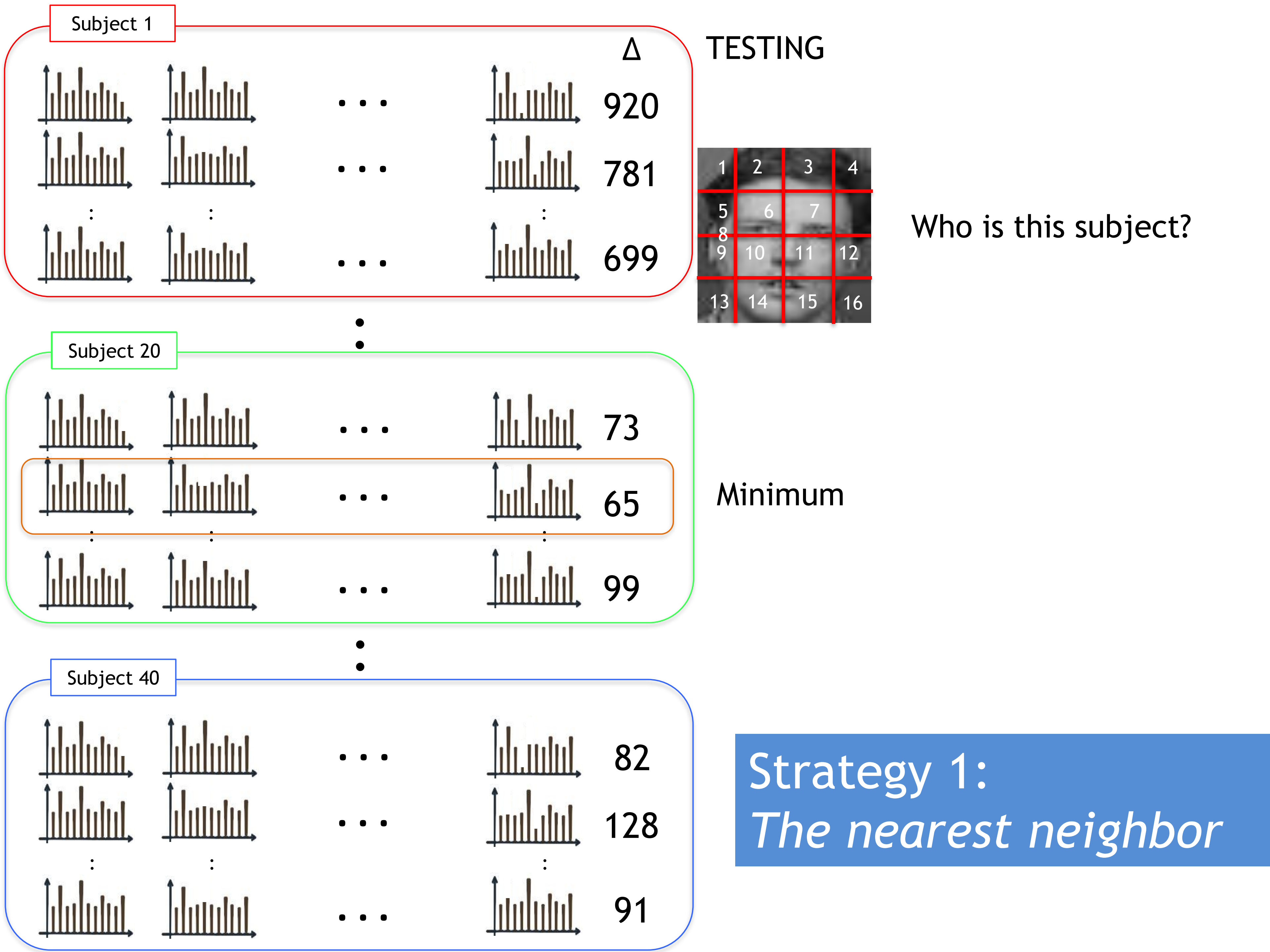
Subject 1
Subject 20
Subject 40
TESTING
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Who is this subject?



TESTING

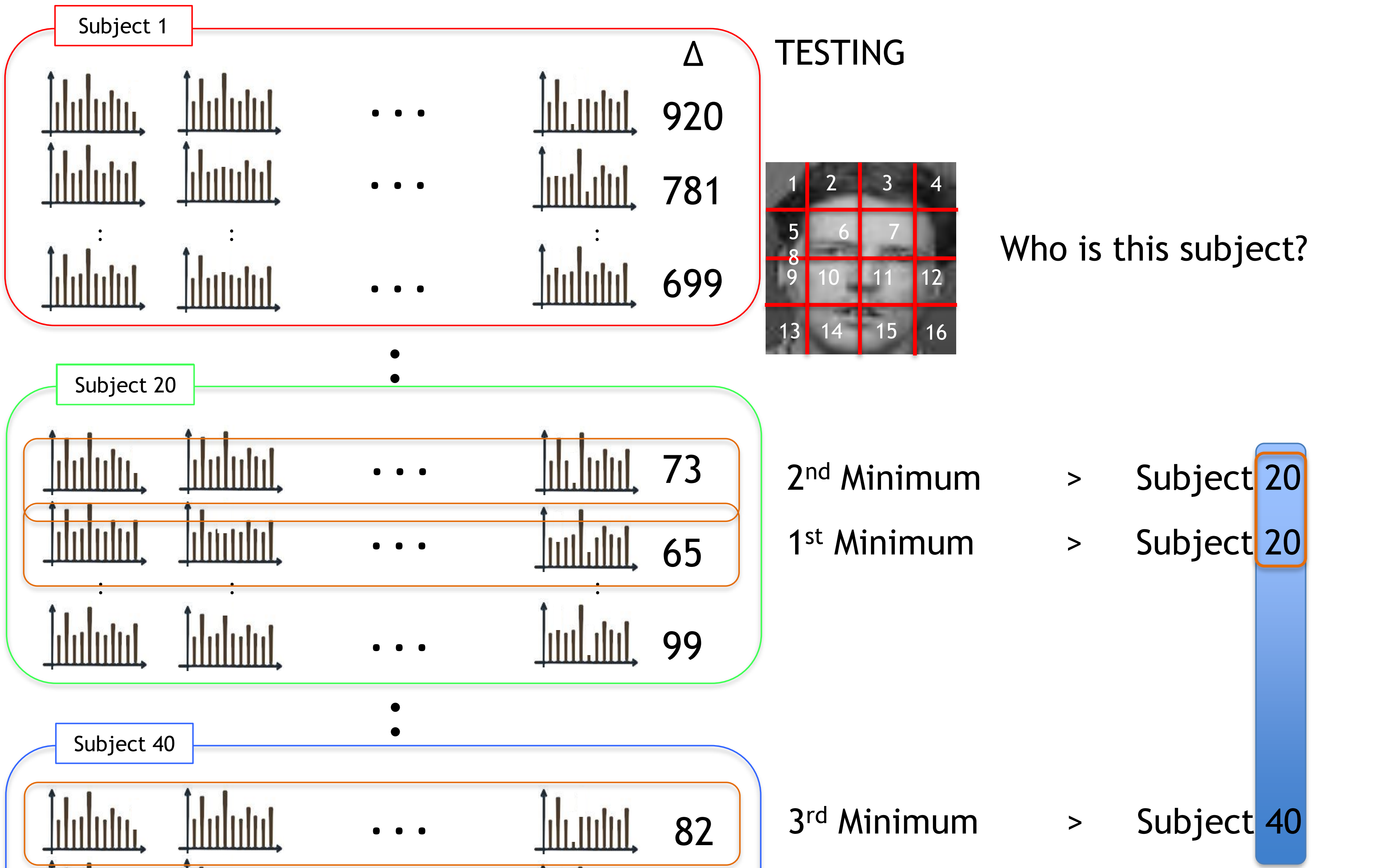


Who is this subject?



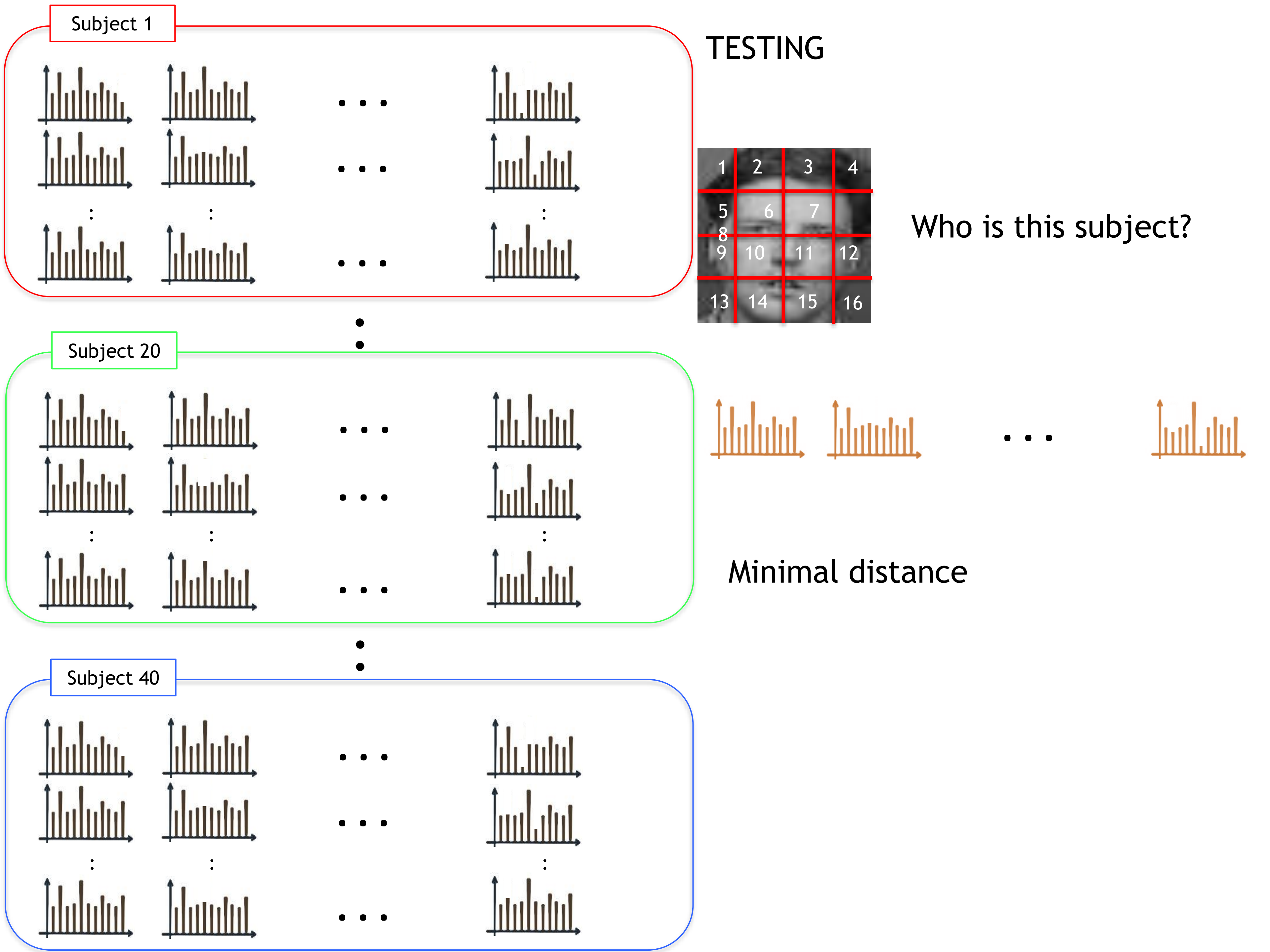
Subject 1
Δ
TESTING
920
781
699
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Who is this subject?
Subject 20
73
65
99
Minimum
Subject 40
82
128
91
Strategy 1:
The nearest neighbor

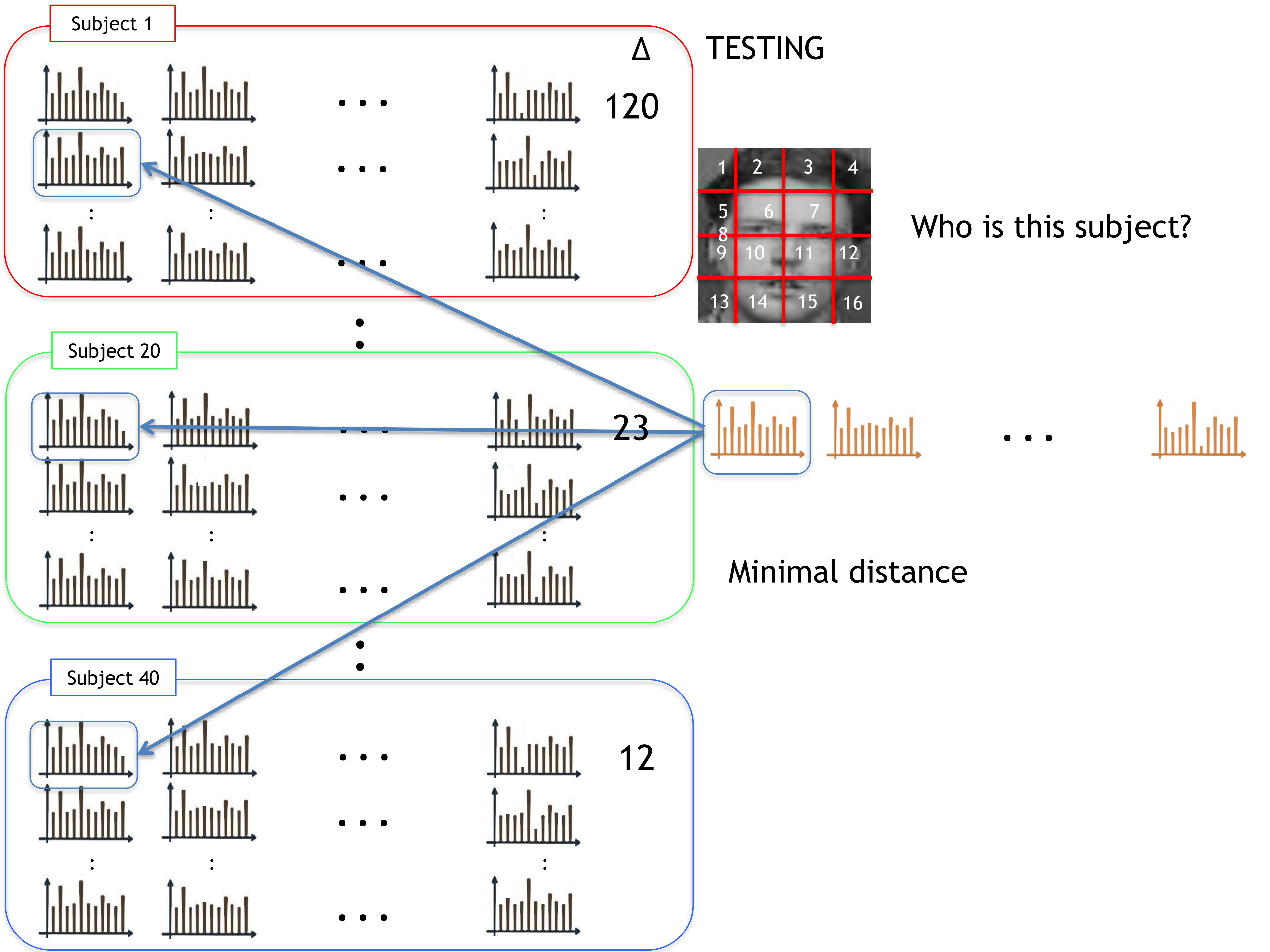
Strategy 2:
k – nearest neighbors
(knn)

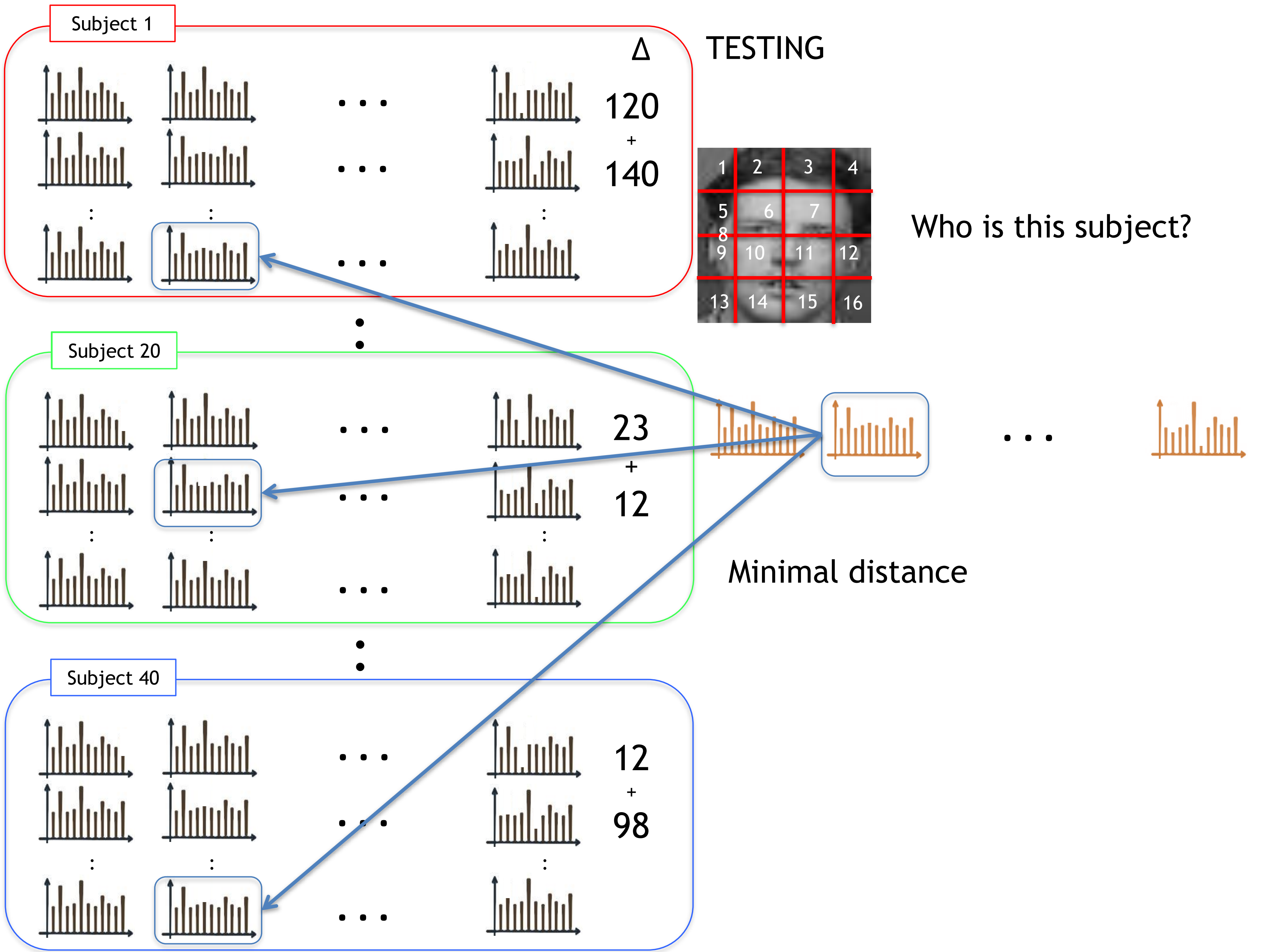


Strategy 2:
k-nearest neighbor (k=3)

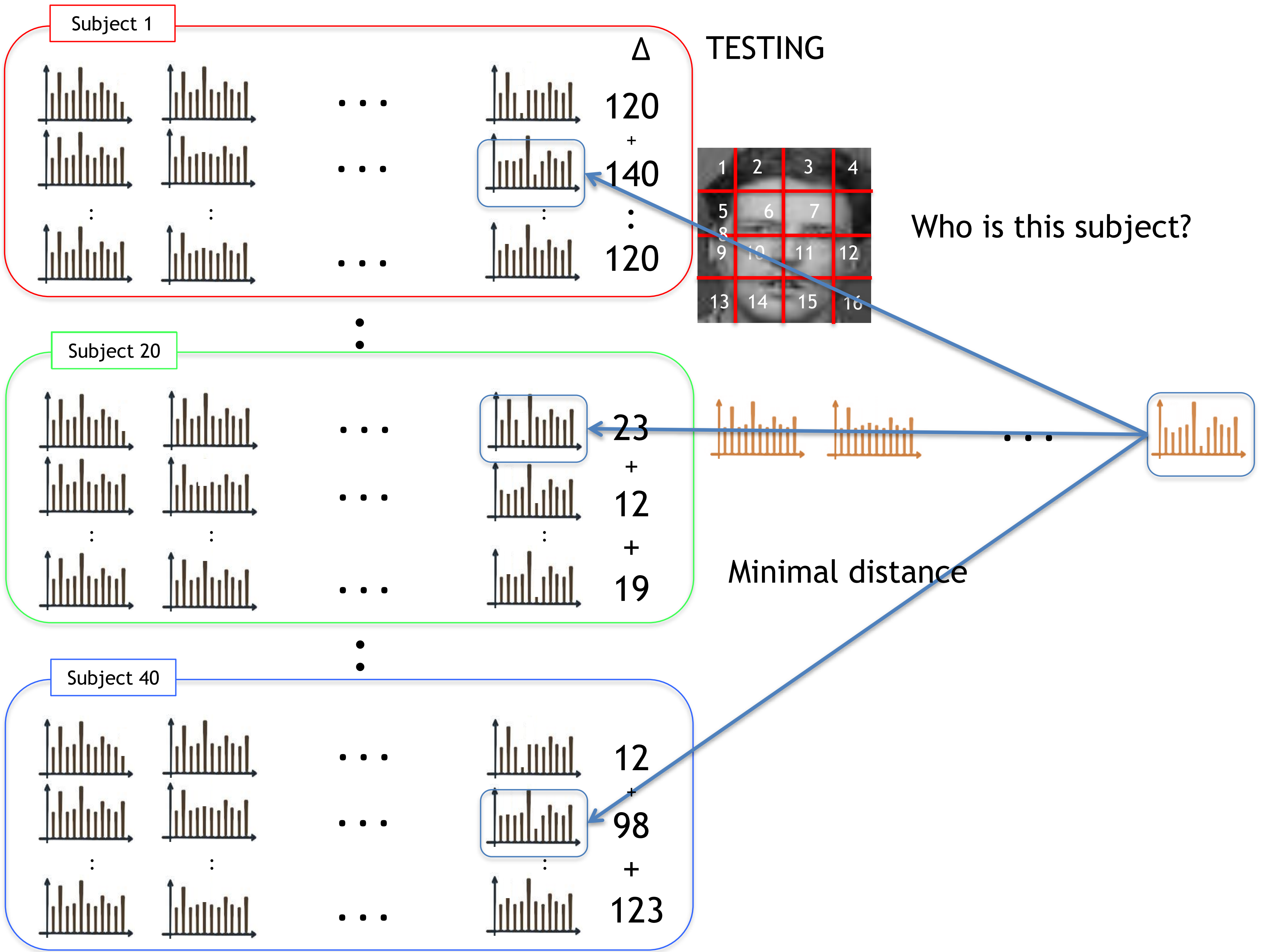
Strategy 3:
smallest sample-class distance

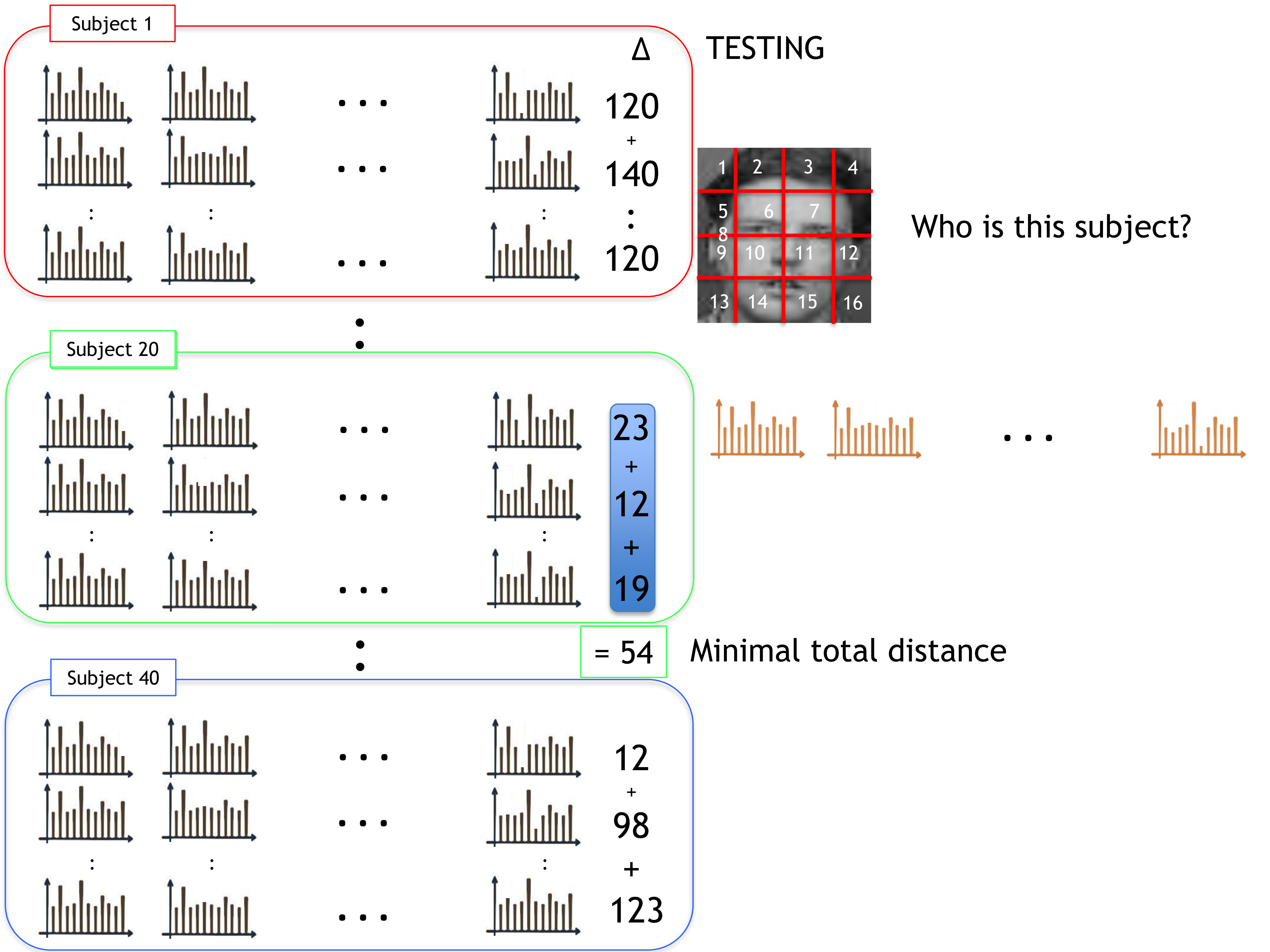

Subject 1
TESTING
Who is this subject?
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Subject 20
Minimal distance
Subject 40


Subject 1
Δ
TESTING
120
1 2 3 4
5 6 7
8
9 10 11 12
13 14 15 16
Who is this subject?
Subject 20
23
Minimal distance
Subject 40
12



Subject 1
Δ
120
+
140
TESTING
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Who is this subject?
Subject 20
23
+
12
Minimal distance
Subject 40
12
+
98


Subject 1
Δ
TESTING
120
+
140
:
120
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Who is this subject?
Subject 20
23
+
12
+
19
Minimal distance
Subject 40
12
+
98
+
123



Subject 1
Δ
TESTING
120
+
140
:
120
1 2 3 4
5 6 7
8
9 10 11 12
13 14 15 16
Who is this subject?
Subject 20
23
+
12
+
19
= 54
Minimal total distance
Subject 40
12
+
98
+
123